AD-A047 615 COMPUTER SCIENCES CORP FALLS CHURCH VA F/G 17/2.1

TDMA SATCOM SYSTEM SIMULATION.(U)

NOV 77 G FRENKEL, P KELL F30602-76-C-0247

UNCLASSIFIED RADC-TR-77-354 NL

1 OF 2

AD

A047615

ADA047615

RADC-TR-77-354
Final Technical Report
November 1977

TDMA SATCOM SYSTEM SIMULATION

Computer Sciences Corporation

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, New York 13441

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-77-354 has been reviewed and is approved for publication.

APPROVED: Stuart H. Talbot

STUART H. TALBOT
Project Engineer

APPROVED: Fred I. Diamond

FRED I. DIAMOND
Technical Director
Communications and Control Division

FOR THE COMMANDER: John P. Huss

JOHN P. HUSS
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (DCCR), Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return this copy. Retain or destroy.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-77-354	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) TDMA SATCOM SYSTEM SIMULATION.		5. TYPE OF REPORT & PERIOD COVERED Final Technical Report,
		6. PERFORMING ORG. REPORT NUMBER N/A
7. AUTHOR(s) G. Frenkel, P. Kell		8. CONTRACT OR GRANT NUMBER(s) F30602-76-C-0247
9. PERFORMING ORGANIZATION NAME AND ADDRESS Computer Sciences Corporation 6565 Arlington Blvd Falls Church VA 22046		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS J.O. 65230914
11. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (DCCR) Griffiss AFB NY 13441		12. REPORT DATE November 1977
		13. NUMBER OF PAGES 111
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Same		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer: Stuart H. Talbot (DCCR)

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Satellite System, Time Division Multiple Access, Demand Assignment, Simulation, Software

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A flexible system simulation program was developed that provides the capability of investigating and demonstrating the operational potential achievable when the RADC developed TDMA technology is applied to a wide variety of satellite or relay communications requirements. The systems to be simulated consist of a large number of low-duty-cycle communication links. When a user needs to establish a link he requests TDMA time slots from a Network Control Terminal (NCT) through low data rate orderwire channels (linking slots) which are either

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

405 717

permanently assigned to user, or are randomly accessed. The NCT uses a sophisticated allocation algorithm and sends back to the requester the allocated slot(s), if slots are available. The frame rate format (linking and communication slots) and slots per frame is flexible. Up to 2500 users and 1250 links can be simulated in any connectivity. For each link the data rate, burst rate, and average number of calls per second can be separately specified. Eight priorities can be simulated for any call. Each user is separately simulated in terms of his "state", e.g., waiting for a channel assignment, establishing contact with the destination user, etc. Various user reactions are also involved. The arrival time of calls is Poisson with the distributed parameter determined by the average number of calls per link. Call durations are exponentially distributed. The software simulates frame-by-frame, the orderwire messages from users to NCT and from NCT to users in a buffer which is the interface between the two halves of a software namely the Network Control Software (NCS) which simulates the NCT allocation algorithm, and the Simulation Software (SS) which models the user. Appropriate statistics are gathered for each run, e.g., average waiting time, blocking probabilities, average frame occupancy, etc.

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont'd)

LIST OF ILLUSTRATIONS

LIST OF TABLES

EVALUATION

The objective of this effort was to develop a flexible system simulation capability to evaluate the performance of existing or planned TDMA satellite or relayed communications systems. The simulation software allows the user to model a number of the parameters of the communication system under investigation including the frame format, signalling discipline, connectivity, traffic intensity, slot assignment strategy, and user priorities. At the end of each simulation run, a statistical summary of 12 categories of activities completed provides a figure of merit relating to how well a particular system design satisfies the requirements. In developing such a capability, RADC has provided the DOD satellite communications community a valuable tool that evaluates achievable service as a function of user requirements.

The results of this effort represent a part of an integrated RADC effort for advanced communications techniques under TPO-10. The technology developed is applicable to the planning and design of future satellite or relayed communications systems.

Stuart Talbot
STUART TALBOT
Project Engineer

1. INTRODUCTION

This document is the final report on " TDMA SATCOM System Simulation," a contract performed by Computer Sciences Corporation (CSC) for Rome Air Development Center (RADC) under contract No. F30602-76-C-0247.

A series of efforts initiated by RADC since 1968 have resulted in concepts, hardware, and supporting software which represent an effective technological base required for the implementation of a TDMA Demand Assignment (DA) system. Major outputs of these efforts included:

- Fabrication of prototype TDMA modems.
- Development of DA network control concepts and their translation into Network Control Software (NCS).
- Development of Simulation Software (SS) for a fully loaded satellite communication network and its user community.
- Development of adaptive null steering arrays.

The DA techniques developed through these efforts were sufficiently broad for application to any multiple-access DA satellite system. Nevertheless, implementation of these techniques was limited to a specific configuration which consisted of the choice of one of two frame formats, two data rates, a fixed signaling structure, and NCS and SS parameters that were uniquely matched to the characteristics of the modem.

There was recognition of the desirability of expanding the capabilities of the NCS and SS to emulate a broad spectrum of satellite system parameters. Hence, the TDMA SATCOM system simulation effort was initiated by RADC. Broadly, the objective, defined in the RADC statement of work, was to develop a truly flexible quick-reaction capability which would allow the evaluation of any existing or planned system configuration. The software system developed

through this effort, referred to as the TDMA SATCOM System Simulation Software (TSSS), represents a powerful tool of high utility.

The software system is capable of evaluating, in numerical terms, the performance of existing or planned DA satellite communication systems through the modeling of the parameters of the system to be investigated (frame format, signaling discipline, terminal G/T, connectivity, data rates, traffic intensity, etc.) as well as the specific DA network control techniques of interest.

A companion of this report is "TDMA SATCOM System Simulation Software Documentation," issued in May 1977. It contains the detailed description and flow charts of the TSSS as well as a user's guide. The TSSS was developed at CSC's Falls Church facility, and runs on the Honeywell 6180 computer at RADC. For the development, a Decwriter II Terminal was installed in Falls Church and connected through a dedicated telephone line to the ARPANET TIP at Reston, Va., and hence to the Honeywell computer at RADC.

2. TDMA DEMAND ASSIGNMENT SYSTEM AND SIMULATION PARAMETERS

2.1 BASIC CONFIGURATION

2.1.1 Method of Demand Assignment and Multiple Access

The throughput and operational efficiency of current Satellite Communications systems has been limited primarily because of inefficient multiple-access and system control strategies. This has led to the development and employment of demand assignment techniques that have increased trunking efficiency by assigning transponder communcation capacity on a real-time demand basis rather than on a preassigned basis.

The primary advantage of demand assignment over preassignment is that the preassigned circuit consumes satellite capacity even during idle periods on its circuit, whereas the demand assignment circuit releases any satellite capacity it has been using for other users. Basically, the demand assignment network is analogous to a telephone switching office that permits many users to share a limited number of trunks and common equipment.

Initially, operational commercial telephone trunks over satellites were assigned the necessary frequencies, power, and bandwidth to provide full-time, point-to-point circuits (e.g., United Kingdom to South Africa, United States to Brazil). The cost of satellite communications ruled out full-time circuits for low traffic volume requirements, such as Nigeria to Peru.

A demand assignment system will allow any user to establish a circuit with any other network station on a demand basis. When the call is completed, the stations involved release the portion of the satellite transponder capacity they were using to other stations. Obviously, the grade of service and number of simultaneous calls that can be made depend upon the system parameters. However, such a system does take advantage of the higher efficiency of a large trunk group operation as compared to that of numerous small trunk groups.

It also takes advantage of the noncoincidence of busy hours that is inherent in a satellite network covering many time zones.

The primary types of demand assignment systems are illustrated in Figure 1 and described below:

1. Variable source. Each terminal has a number of permanently assigned downlink (receive) channels. The terminals compete for the uplink (transmit) channels. These uplink channels are used on a shared (demand assignment) basis.

2. Variable destination. Each earth terminal has a number of permanently assigned uplink (transmit) channels. The terminals compete for the downlink (receive) channels. These downlink channels are used on a shared (demand assignment) basis.

3. Fully variable. In this case none of the channels involved are permanently assigned to any earth terminal. Thus, any station can use any channel to call any other station, provided the channel is not already in use.

Fully variable demand assignment is the most efficient of the three types, but it requires more sophisticated control. This report concentrates on the fully variable technique. It is the most applicable demand assignment strategy for tactical systems involving large numbers of low duty cycle users.

The method whereby a portion of the satellite resources is assigned to a call depends on the multiple-access technique. The two such techniques particularly suited for demand assignment are FDMA, of which the SPADE system is the best publicized example, and TDMA. In a TDMA demand assignment system, a call is serviced by allocating a time slot or a set of time slots within the TDMA frame format. If the terminals do not have equal figures of merit, to ensure equal voice quality, the fraction of satellite power dedicated to the one-way voice channel is decreased with the figure of merit of the receiving terminal. This is accomplished by assigning a different number of time slots to a one-way voice channel according to the figure of merit.

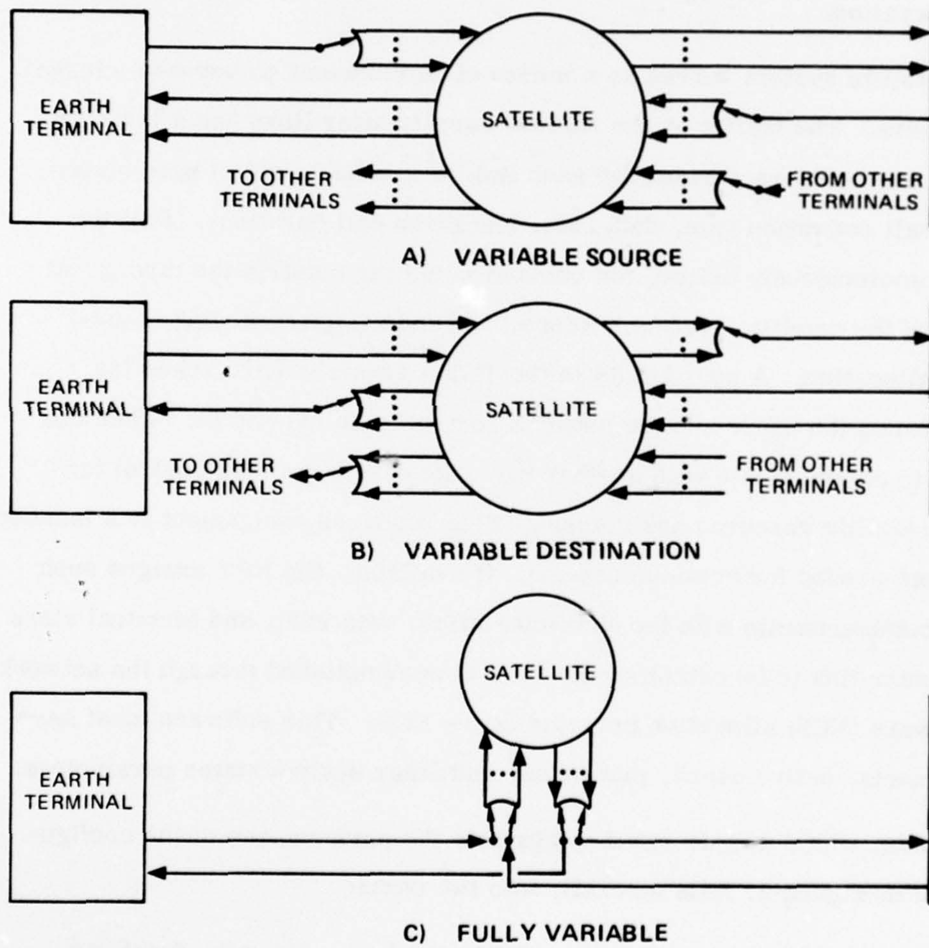


Figure 1. Types of Demand Assignment

TDMA offers higher performance and more flexibility than FDMA. Since only one carrier is present in the transponder, there is no power loss or interference due to intermodulation. The power output device can be run at saturation, thus providing higher output power (typically 3 dB) and efficiency. There is no need for earth terminal power control.

2.1.2 Overview of DA Systems and Simulation

Figure 2 illustrates the structure of the satellite communication system under consideration.

The satellite system serves as a means of communication between a large number of users. The traffic on the various user-to-user links has a low duty cycle which may be characterized for each link by a suitable set of parameters, such as the call activation rate, data rate, and mean call duration. If all the links were simultaneously active, the demand would far outstrip the throughput capabilities of the satellite system. Hence, demand assignment (DA) is used for channel allocation. A set of slots in the TDMA frame is earmarked for signaling between the users and the network control terminal (NCT). When one user wishes to communicate with another user, he asks a network control terminal for a satellite resource assignment. This resource assignment is a number of TDMA slots needed for communications. If available, the NCT assigns such a resource commensurate with the characteristics, data rate, and terminal sizes of the particular link to be established. This is accomplished through the network control software (NCS) allocation program at the NCT. This software must keep track of channels, active users, priorities, and other system status parameters.

The TSSS, which was designed to evaluate the performance of the configuration shown in Figure 2, falls naturally into two parts:

1. The Network Control Software (NCS) which simulates the functions of the NCT, such as the allocation algorithms, control functions, user tables, TDMA channel assignment tables, and signaling from the NCT to users.

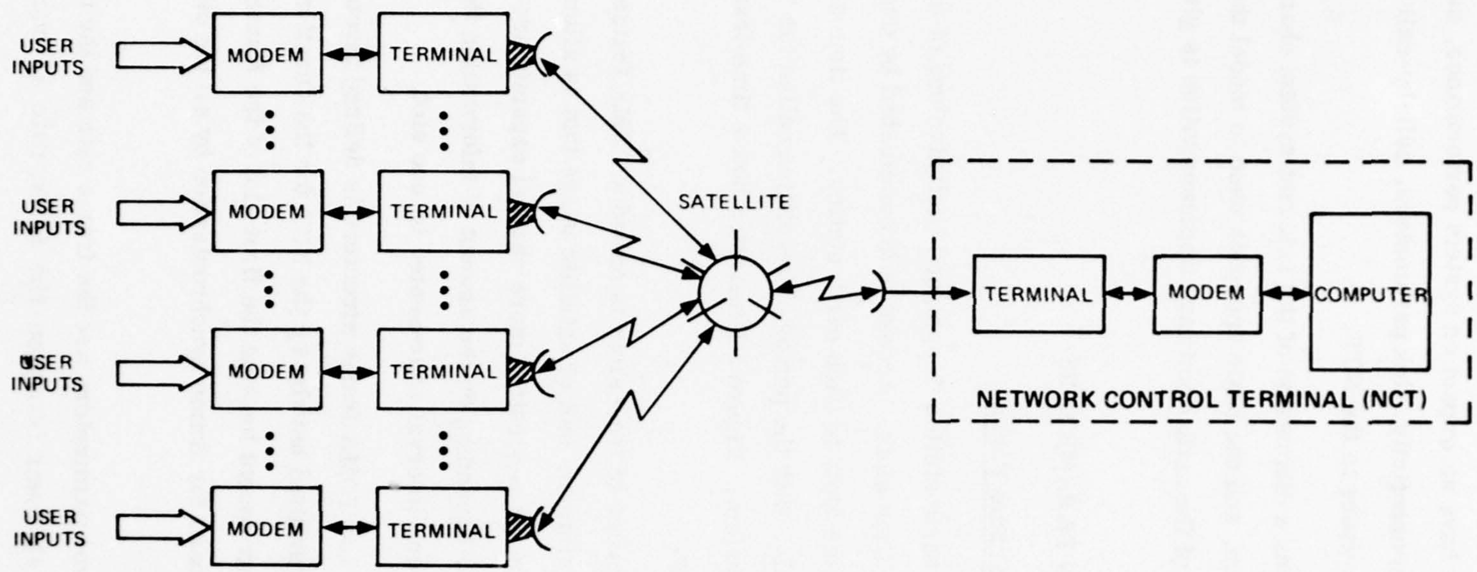


Figure 2. TDMA Demand Assignment System

2. The Simulation Software (SS) which models the user community and its features which have an impact on system performance, such as user-to-user connectivity, link parameters, call-by-call history, and signaling from users to the NCT.

This section contains a discussion of the relevant system characteristics, their impact on simulation, and the basic approach used to model these characteristics. A discussion of the actual software implementation is given in Section 3.

2.2 TDMA SIMULATION PARAMETERS

2.2.1 Frame Rate and Frame Format

With TDMA, each earth station is assigned exclusive use of the satellite repeater during specific time slots. Access is accomplished by designating a particular sequence of time slots to each earth station. The time slots for a channel occur periodically, with the period of repetition called the "frame repetition rate" of the system. Figure 3 illustrates how a time frame might be divided into time slots.

Each periodic sequence of time slots is called a TDMA frame. Each individual slot represents the minimum unit of satellite usage that is allowed in the TDMA system. Those users requesting more channel capacity are assigned an integral number of slots, depending on the amount of information that can be transmitted during the time interval represented by one slot.

The time base for the TDMA frame structure is defined through a synchronizing signal that is transmitted usually by the NCT for the duration of one time slot which by common agreement became the first slot of the frame (Figure 3). This time reference is used for frame synchronization by all the other users of the TDMA frame.

The two basic frame parameters are the frame rate and the number of slots per frame. A practical lower bound on the frame rate is one frame in 3 to 4

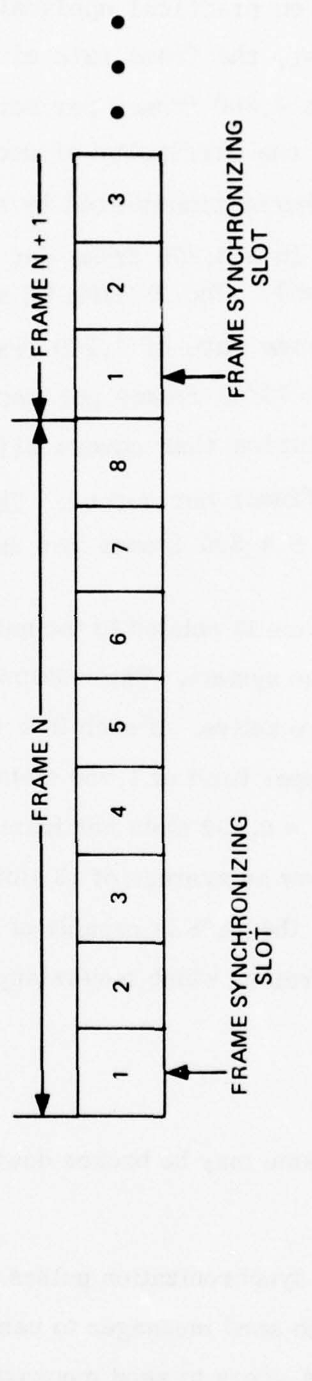


Figure 3. Frame Format

seconds. Any lower frame rate creates serious time delay problems with associated stress on the talker using a telephone channel. It also results in large buffer memory requirements, especially for high data rate users. The upper limit of the frame rate based on practical applications is 4,800 frames per second. As a point of interest, the frame rate of the AN/USC-28 spread spectrum modem for DSCS-III is 4,800 frames per second. The primary objective of higher frame rates is the difficulty of accommodating low data rate users. For example, if each burst transmitted by a user contains only 20 bits, the resulting data rate in a 4,800 frame per second system would still be 96 Kbps (20 x 4800 = 96 Kbps). The PN/TDMA HC modem developed by Raytheon for DSCS III has a frame rate of 1,200 frames per second; the frame rate of the RADC TDMA modem is 75/32 frames per second. In conclusion, a range of frame rates for simulation that covers all rates to be encountered in practice is 0.25 to 4,800 frames per second. The TSSS uses a range of $75 \times 2^{-14} = 0.0045$ to $75 \times 2^{6} = 4,800$ frames per second, which covers the above range.

The number of slots per frame is related to the number of links that can be supported simultaneously by the system. The maximum number of links simulated is 1,250, of which not all are active. If each link would utilize only a single slot, this would yield an upper limit of 1,250 slots per frame. The TSSS is capable of simulating up to $2^{13} = 8,192$ slots per frame. If half the links are active at one time, this would allow an average of 13 slots per link, which is more than ample. In conclusion, the TSSS is capable of simulating a range of $2^{4} = 16$ to $2^{13} = 8,192$ slots per frame, which covers any value likely to occur in practice.

2.2.2 Slot Structure

The TDMA slots in each frame may be broken down into the following categories:

- Slots containing frame synchronization pulses.
- Slots used by the NCS to send messages to users.
- Slots used by the TDMA users to send messages to the NCS.
- Slots used by the TDMA users to communicate between themselves.

The slots for frame synchronization may be regarded as an overhead since they cannot be available for user-to-user communications. Other overhead requirements include slots reserved for the transmission of user information to the NCT and the transmission of NCT information to particular users. Slots occupied in this way are called linking slots. The manner in which the linking slots are used will be discussed in Paragraph 2.3.1. The remaining slots in the TDMA frame can be used for communication between users.

The number of TDMA slots needed to establish a link between two users is obtained from the link burst rate and data rate, as discussed in Paragraph 2.5.1. It is rounded off upward to the power of two. A type-n channel is defined by the

$$\text{Number of slots per channel} = 2^{\text{total number of slots in TDMA frame-type}} \text{ channel.}$$

An important consideration in network control is the location within the TDMA frame of the TDMA slots for a particular channel that establishes a link. There are three alternatives as illustrated in Figure 4:

1. TDMA slots for the same channel are adjacent within the TDMA time frame.

2. TDMA slots for the same channel are dispersed and equally spaced within the frame.

3. TDMA slots for the same channel are randomly dispersed within the TDMA frame.

The first two alternatives are identical in performance. In a frame with 2^s slots, the identity of the 2^{s-n} slots of a type-n channel is determined in the first method by an s-bit word and all possible combinations of the last s-n bits. In the second method, all possible combinations of the first s-n bits determine the identity of the 2^{s-n} slots. Given any scenario in one method, an exactly identical scenario can be written with the same performance, occupancy, and availability by reversing the order of bits in each channel ID. Since these two methods are identical in performance, there is no point in simulating both.

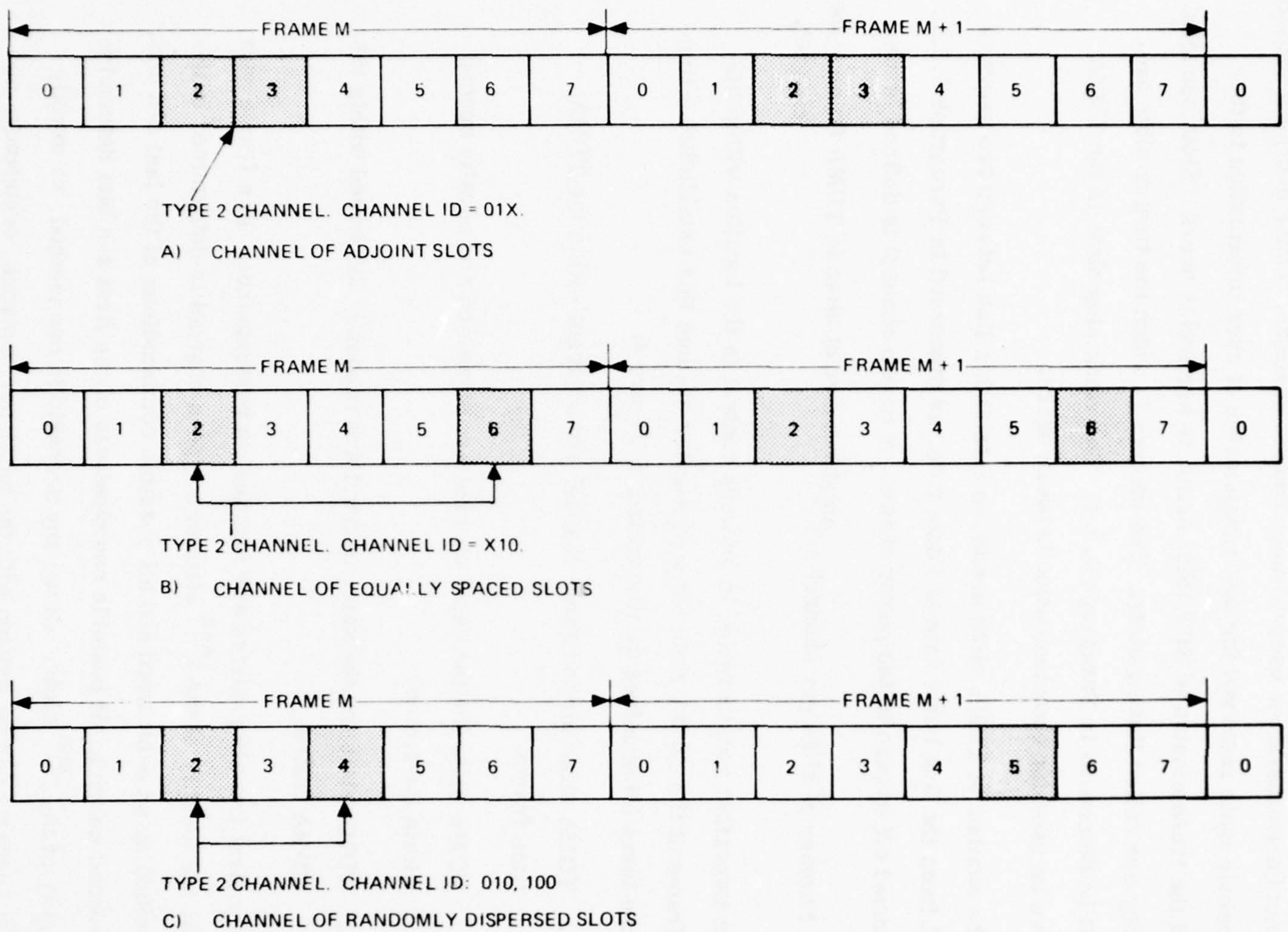


Figure 4. Methods of Channel Assignment

The drawback of the first two methods is that they are somewhat restrictive. By referring to Figure 4, it is possible to have two slots still available (not assigned) within the frame; but, if they are not adjacent, and if the first method of assignment is used, a type 2 channel cannot be assigned.

The third method is more flexible, from the viewpoint of availability, but each slot must be separately designated by an ID number. This makes signaling and control cumbersome. Since the first two methods are equivalent, and the third one is unpractical, the TSSS uses the first method of channel assignment in the NCS.

In a system with 2^s slots in the TDMA frame, the type of channel may range from 1 to s. Since the TSSS has a maximum s-capacity of 13, or 8,192 slots per frame, the channel type in the TSSS may vary from 1 to 12, but it may not exceed s for any specific run.

Having defined the method of designating the ID for the various types of channels, the TDMA frame structure may be readily defined. A typical example is shown in Figure 5. It has the following parameters:

Number of slots per frame is $2^s = 512$ or $s = 9$

Slots	Type	ID	Slot No.
Synchronizing	8	00000000X	0-1
Linking (NCS to user)	8	00000001X	2-3
	7	0000001XX	4-7
	6	000001XXX	8-15
	5	00001XXXX	16-31
Linking (users to NCS)	4	0001XXXXX	32-63
	4	0010XXXXX	64-95
Communications	4	0011XXXXX	96-127
	5	01XXXXXXX	128-157
	6	1XXXXXXXX	258-511

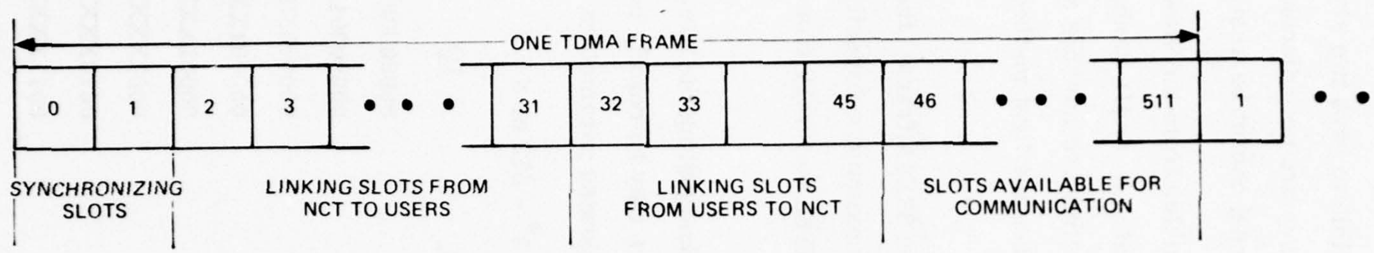


Figure 5. Typical Slot Structure

Of course any type-n channel may be subdivided into two type n+1 channels, four type n+2 channels, etc.

2.3 DEMAND ASSIGNMENT SIMULATION

2.3.1 Linking Schemes

The transfer of information between the NCT and the user is an essential part of a DA system based on centralized control. The terms "order wire," "linking," and "signaling" are used interchangeably for this operation. The TDMA slots reserved for user to NCT signaling may be used in one of three ways:

1. Permanently and exclusively assigned linking slots to each user. The advantage of this scheme is the lack of mutual interference between users. The drawback is that too many slots must be assigned if the number of users is high, thus increasing the overhead and causing loss of efficiency.

2. Polling. This method ensures more flexibility than permanently assigned linking slots. However, the overhead is potentially high since each message requires two-way transmission. As in the case of permanently assigned linking slots, mutual interference between users is eliminated.

3. Random access of linking slots. The number of linking slots can be much smaller than the number of users; hence, potentially, the overhead is reduced, when compared to the other schemes. The drawback is mutual interference between users. In order to favor the higher priority calls, the linking slots may be subdivided into "pools" of linking slots, with one pool for each priority. By assigning a higher ratio of linking slots to higher priority users, the probability of loss of signaling message for these users is reduced.

The methods of transmission of messages within the linking slots can vary greatly. From the point of view of DA performance, there are two key parameters:

1. Probability of message transmission error.

2. Probability of mutual interference between user messages.

The probability of transmission error depends on the link parameters, the method of modulation, and the coding. It is beyond the scope of the type of simulation discussed here to simulate all these factors; however, the message loss itself must be simulated. For example, if the system to be evaluated uses coding, it is straightforward to compute the message loss probability on the coded system and, for each message, to cause a loss of message with the computed probability through a Monte-Carlo approach (random number generator). In this manner by merely setting the probability of message loss to the derived value, the impact of the performance of any order-wire channel (coded or uncoded) may be included. This is, in fact, the approach used in the TSSS.

The most elementary method of coding is the repetition of messages a certain number of times, say M, and validation by ascertaining that N out of these are identical. In this N out of M strategy, the performance depends on the two parameters M and N through various factors, such as probability of message loss, waiting time, and mutual interference in random access mode. The TSSS has full flexibility in the simulation of the parameters M and N. Each message is transmitted M-times and a validation of N out of M is performed, with the probability of error on each of the M-messages at the preset value.

In the polling mode a particular message can be lost because of the loss of either the interrogation from NCT to user or in the opposite direction. The actual probability of message loss is approximately the sum of these two probabilities. In the other two modes (random access and permanently assigned linking slots), the probability of message loss depends only on the one-way (user-to-NCT) message loss. The TSSS includes the Monte-Carlo simulation of message loss for either of these two effects.

The user-to-NCT messages may also be lost because of mutual interference in the random access mode (for linking slots). This effect is included for the TSSS.

A buffer simulates the linking slots of a frame, and the users select a slot at random and insert messages. If more than one user selects the same slot during a frame, the message is considered lost.

At this point, it is convenient to summarize the signaling and error performance simulation features included in the TSSS.

Methods of Linking:

- Permanently assigned slots.
- Polling.
- Random access, with optional number of linking slots assigned to each priority message.

Transmission Error Rate per Message:

Fully variable from 0 to 1.

Transmission Strategy:

Each message transmitted M-times must be received without error at least N-times for validation. M and N fully variable.

Mutual Interference of User Messages:

Simulated through frame-by-frame simulation of linking slots and their messages.

2.3.2 ARQ

This is a method feedback error control in which the controller, when he detects a transmission error, asks the sender for a repeat. The identity of the sender in this case is known only in linking schemes using permanently assigned linking slots or polling. In the random access mode, since users compete for the same linking slot, there is no way of knowing who has sent a message, once the message is lost. The TSSS simulates the validation procedure of the NCT, and ARQ is included as a simulation option. When a user receives an ARQ message, the previous message is sent again and must be validated by the NCT.

2.3.3 Message Structure and Signaling Functions

An examination of the message structure between users and NCT (Table 1) is useful in highlighting the DA functions simulated by the TSSS. Two types of messages are sent between the network controller and the users. Those that are initiated by the user mainly request and relinquish channels. Those that originate with the network controller assign channels, bump users for higher priority users, and query users in order to maintain the network.

Four types of messages are transmitted by the user. When the user needs a channel, he sends a request for it. When the user is finished with the channel, he sends a relinquishment of channel message. If after a certain length of time no channel has been assigned, the user requests his position in the channel assignment queue. The fourth message from the user is a reply to the network controller's status query. The channel type is relevant only when the user requests a channel. Table 1 shows the structure of messages that appear in the TSSS. Parts (5), (6), and (7) are used to simulate transmission errors, mutual interference, and message validation, and they do not have a counterpart in an operational system.

The network controller replies with a channel number when he assigns a channel to the user. If the channels are busy and priorities determine who receives the channels, the controller may bump lower priority users by sending either a request to disconnect immediately or in a preassigned number of seconds. The time delay tau is an initialization parameter. If it is defferent from zero, two messages are sent: the first with 1 in location (2) (see Table 1), the second with -1. The reaction of the user to these messages is discussed further in Paragraph 2.5.

Table 1. Message Structure Between Users and NCT

<u>Message Structure to Network Control Terminal (NCT)</u>

(1) = Userid

(2) = Priority of message

(3) = Channel type or user status

(4) = Message number

1 - Request for assignment

2 - Relinquishment of channel

3 - Request position in assignment queue

4 - User status

(5) = Message count

(6) = Random access user counter

(7) = Validation counter

<u>Message Structure from Network Control Terminal (NCT)</u>

(1) = Userid

(2) = 1, disconnect tau delay

= -1, disconnect no delay

(3) = -1, assignment of channel

= + value, position in assignment queue

(4) = -2, ARQ

= -1, request user status

2.4 CONTROL STRATEGIES

2.4.1 Preempt Options

When a user requests a channel assignment and no suitable channel is available, another user of lower priority may be requested to relinquish his channel. This is done in order to make room for the higher priority user. This preempting process can occur in one of several ways:

- The lower priority user may be forced automatically to disconnect when a channel preempt is indicated by including such a hardware feature into the modems.
- The lower priority user may have the option of not relinquishing if the above-mentioned hardware feature is absent.

In the TSSS, either one of these conditions is established at the beginning of a simulation run by setting a variable to a 1 or a 2. A 2 indicates that the forced channel preempt strategy will be in effect throughout the simulation run. A 1 indicates that the users which are requested to relinquish will have the option of not relinquishing.

A user may be given a finite period of time before being required to disconnect. In this case a preempt warning is first sent to the user, with a final disconnect message sent after tau seconds. This alert for system preempt will allow the user to speed up his transmission in an effort to complete the call in the remaining time. At the completion of tau seconds, either the option of relinquishing or not relinquishing is open to the user, depending on initialization.

A total of eight priorities are simulated in TSSS. The priority is a property of the call; i.e., a particular user may send out a request for different priority channels to the NCT at different times. This is discussed further in Paragraph 2.5. To summarize, the following preempt features are included:

Number of Priorities:

Eight. If desired each user can have a different priority requirement at different times.

Disconnect Strategy:

- Forced or optional.
- Immediate or with delay. Delay is variable. If delayed, a warning is sent first.

2.4.2 System Resource Utilization

Part of the time system resources may be used for other than user-to-user communications. These include time required for synchronization pulses, communication from a user to the NCT, and communication from the NCT to a user. System resource usage is represented by slot allocation in a TDMA system. Slots used for synchronization purposes, and those used for communication between users and the NCS are termed "overhead" slots because they do not contribute to the channel capacity of the system, even though they use part of the satellite transponder's communication capability.

Those TDMA slots that are used for communication between users and the NCS are termed "linking slots." These slots must always be reserved in the TDMA frame in order that user requests and NCS responses can always be sent without hindrance. Likewise, other overhead slots might be permanently assigned out of the TDMA frame to users who require a permanently assigned channel.

The remaining slots for user-to-user messages will be assigned according to user demands. This means that the set of communication slots available for demand assignment will vary from frame to frame, as users request and relinquish channels.

The TSSS has full flexibility in determining the TDMA resource utilization. In addition to the usage discussed above, slots may be reserved; i.e., withdrawn from the DA pool for unspecified use (e.g., permanent assignment).

2.4.3 Functions of NCT

The principal role of the NCT is to allocate channels on request, taking into account the priorities of the calls. In order to do this efficiently, a number of key tasks must be carried out:

1. Keep track of all the TDMA time slots and their dispositions, namely:
   - Assigned
   - Available for assignment
   - Excluded from assignment (overhead or permanently assigned)
2. Keeping track of the status of the users (with respect to channel assignment:
   - User has a channel assigned to him with a specific channel ID and call priority
   - User is waiting for assignment of a channel of a specific type and priority
   - User is inactive
3. Keeping track of all pending requests for assignment by three key parameters:
   - Type of channel
   - Priority of call
   - How long ago the request was made
4. Processing all incoming messages from users:
   - Validation of messages
   - Requests for assignment
   - Relinquishments of channels

- Requests for position in assignment queue
- ARQ

5. Withdrawing channels from lower priority users if needed by higher priority users (bumping).

6. Transmitting messages to users:
   - Channel assignments
   - Requests for relinquishments
   - Warning of an impending request for relinquishment
   - User position in assignment queue
   - Request for user status

2.4.4 Channel Assignment Procedure and Theory

The problem of efficient assignment of TDMA time slots in a demand assignment system was analyzed in the predecessor of this effort, "TDMA Network Control Study," performed by CSC for RADC in 1973 (Reference 2). The conclusion was reached that, in a system in which different links require a different number of slots to establish a channel, the primary source of loss of efficiency is fragmentation. This means, in effect, that user requests cannot be satisfied because, although the desired number of slots is available, these are dispersed in random fashion throughout the frame. Unless special measures are taken, fragmentation invariably occurs after a system is in operation for a reasonable amount of time, and steady-state conditions are reached. Fragmentation can be prevented if the channel assignment algorithm is designed to assign channels into the most heavily loaded portions of the TDMA frame. (This assumes that a channel consists of adjacent TDMA time slots.) When this is done, the unassigned TDMA time slots also tend to bunch together, and the probability of finding the desired number of adjacent vacant slots increases. An algorithm was developed and presented in Reference 2. It was designed for the frame format of the RADC modem. The channel assignment algorithm developed for the TSSS is a generalization of the approach, in the sense that it is fully flexible from

the point of view of frame parameters and channel types. Because of the key importance of this algorithm in the overall approach, this paragraph presents it in some detail.

Figure 6 shows a tree diagram which describes the division of a frame into various channel sizes. This diagram is called the channel assignment tree. The number of slots in the TDMA frame is assumed to be a power of two. In this way, the TDMA frame can be successively divided in half until a single slot is obtained. This is the basis for different channel sizes. A channel is defined as any adjacent grouping of a power of two number of slots, which can be obtained by successive half-divisions of the TDMA frame. No channel of an odd multiple number of slots, except for a single slot, can be assigned. A request for an odd number of slots will be assigned a channel containing a power of two number of slots that is just greater than the requested number of slots. In this way, NT different sized channels can be used for channel assignments. The value of NT depends on the number of slots contained in a TDMA frame. Each TDMA frame consists of 2^{NT} slots. NT can be initialized to any integer value between 1 and 12 at the beginning of a simulation run.

A channel type number is used to represent the size of a channel. Larger channel type numbers represent smaller channels. The largest channel that can be assigned consists of half the TDMA frame. The entire frame cannot be assigned, since some slots inevitably will be used for linking purposes. The relationship between the number of slots contained in a particular channel and its channel type number is given as follows:

$$\text{number of slots used by a channel in a TDMA frame} = 2^{NT\text{-type}}$$

where type is the type number of the channel, and NT is the number of different channel sizes. Thus for an NT of 7, a type 3 channel will have 2^{7-3} or 16 slots per frame. The frame has 128 slots.

Each node of the channel assignment tree represents a unique set of slots (channel). Vertically grouped nodes represent channels containing the same

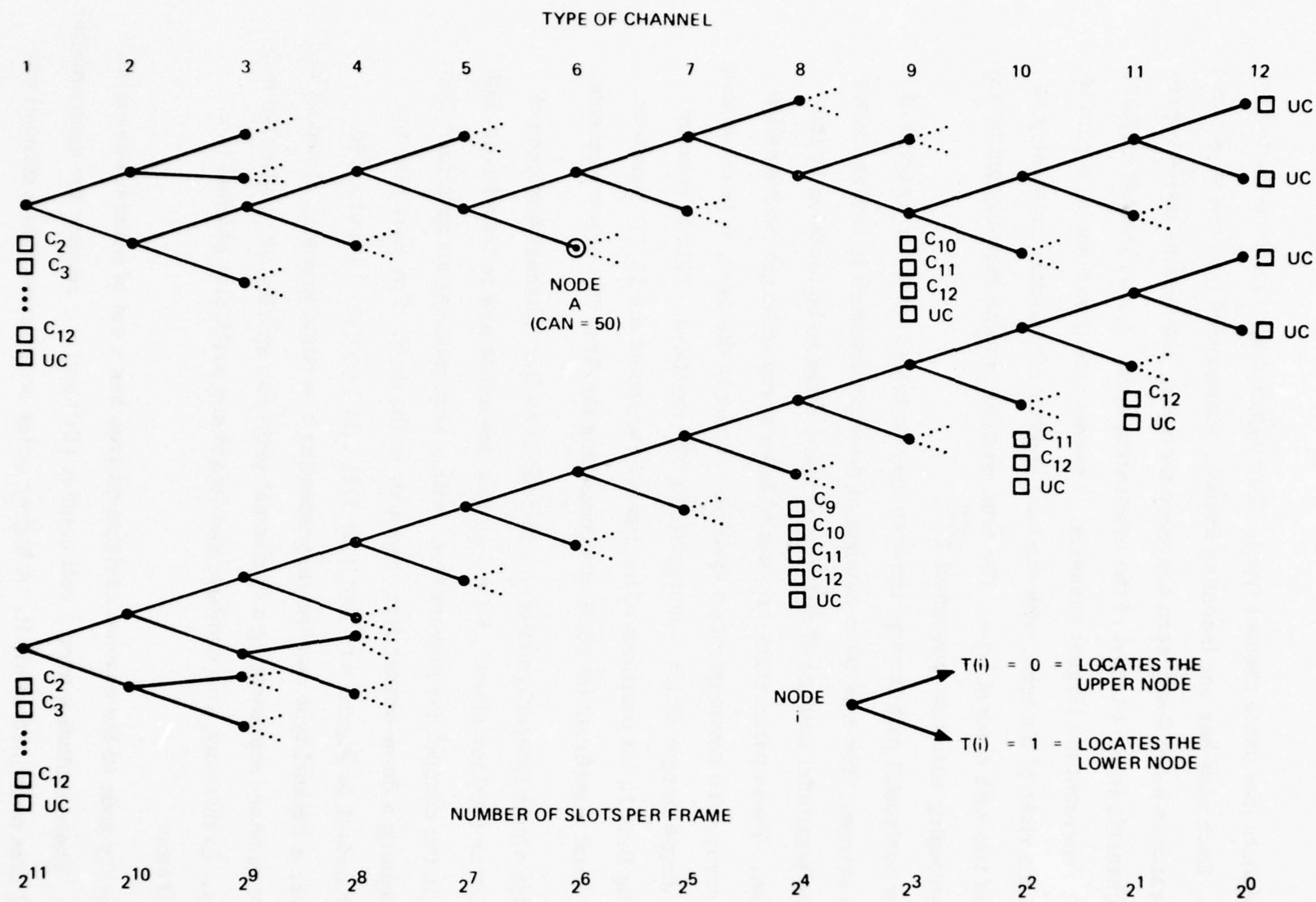


Figure 6. Channel Assignment Tree

number of slots (the same channel type). Two adjacent nodes are connected by a branch. Each node has one incoming branch, represented by a one or a zero. All one branches move down; zero branches move up. Successive nodes represent the channels in the channel of the originating node. Nodes located farthest to the left represent the largest channels. Channel size decreases as a path is traced to the right of the tree, with the total number of channels increasing as a power of two with channel type. The total number of slots represented by any vertical grouping remains constant at 2^{NT}.

Any horizontal path through the tree may be traced by using a series of ones and zeroes. The maximum number of these parameters is twelve, since that is the maximum number of different channel sizes to be modeled by the simulation. These parameters are stored in an array of length twelve called the "T" array. To communicate a specific channel to the user, it is convenient to use a single integer which contains all 12 "T" numbers. This is done by letting the first 12 bit positions of the integer represent the 12 "T" numbers, with the least significant bit positions containing the first "T" array elements.

This single integer quantity is referred to as the "Channel Assignment Number," or CAN for short. A CAN simply describes how to travel from left to right in the channel assignment tree, with a zero meaning an up-branch and a one meaning a down-branch from one node to the next. For example, the CAN for node A in Figure 6 is T(6) T(5) T(4) T(3) T(2) T(1) = 110010 = 50. Together, a channel type number (representing a vertical grouping of nodes) and the CAN number (representing a horizontal path) can specify any particular node uniquely. In this way, two numbers can locate any particular channel in a TDMA frame.

Every node in the channel assignment tree has a set of counters associated with it. These include a user's cell counter (UC) and one counter for each smaller channel size contained within it. A higher order node represents a channel contained in the lower order node (Table 2).

At every node, the counters give information about the availability of the higher type of channels contained within it. Higher channel types are represented by nodes which branch out of that node.

The first counter of each node indicates how many channels of its next higher type are occupied. The following counters contain information about the occupancy of higher channel types contained within the node. When a particular type of channel is assigned (or relinquished), the contents of the corresponding counters are to be incremented (or reduced) by 1. The maximum number in each counter for all types of nodes (channels) is listed in Table 2.

Table 2. Maximum Counters and Content of Nodes

Counters	Node Type 1	2	3	4	5	6	7	8	9	10	11	12
C_2	2											
C_3	2^2	2										
C_4	2^3	2^2	2									
C_5	2^4	2^3	2^2	2								
C_6	2^5	2^4	2^3	2^2	2							
C_7	2^6	2^5	2^4	2^3	2^2	2						
C_8	2^7	2^6	2^5	2^4	2^3	2^2	2					
C_9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2				
C_{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2			
C_{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2		
C_{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2	
UC	User ID or -1											UC

In general, at node type n, C_m can take on a value no larger than 2^{m-n}. Since the next-to-last counter at a node is C_{NT}, this counter has a maximum content which depends upon the node wherein it is located (Table 3).

Table 3. Maximum Counter and Content of C_{NT} Node Types

Counter	Node Type											
	1	2	3	4	5	6	7	8	9	10	11	12
C_{NT}	2^{NT-1}	2^{NT-2}	2^{NT-3}	2^{NT-4}	2^{NT-5}	2^{NT-6}	....					2^{NT-12}

The last counter at a node is called a user cell. It can have one of five contents:

(i) User Identification number (UID)

(ii) -1

(iii) 0

(iv) -2

(v) -3

where

(i) That channel is assigned to a user with user ID number (UID).

(ii) That channel is not available for assignment. (It is either completely occupied by a lower type channel user or partially occupied by a higher type channel user.)

(iii) Empty and available channel.

(iv) Reserved for overhead (e.g., linking slots).

(v) Reserved for other purposes.

Figure 7 shows the construction of the Channel Assignment Table (CAT). This table implements the operation of the channel assignment tree by keeping a record of the status of every channel in the TDMA frame. The method of locating the different counters in the CAT table is now described.

To obtain information about a particular channel, the counters of various nodes must be accessed. For a type 1 node, the desired node is indicated by the value of variable T(1). If T(1) equals zero, the type 1 node associated with the

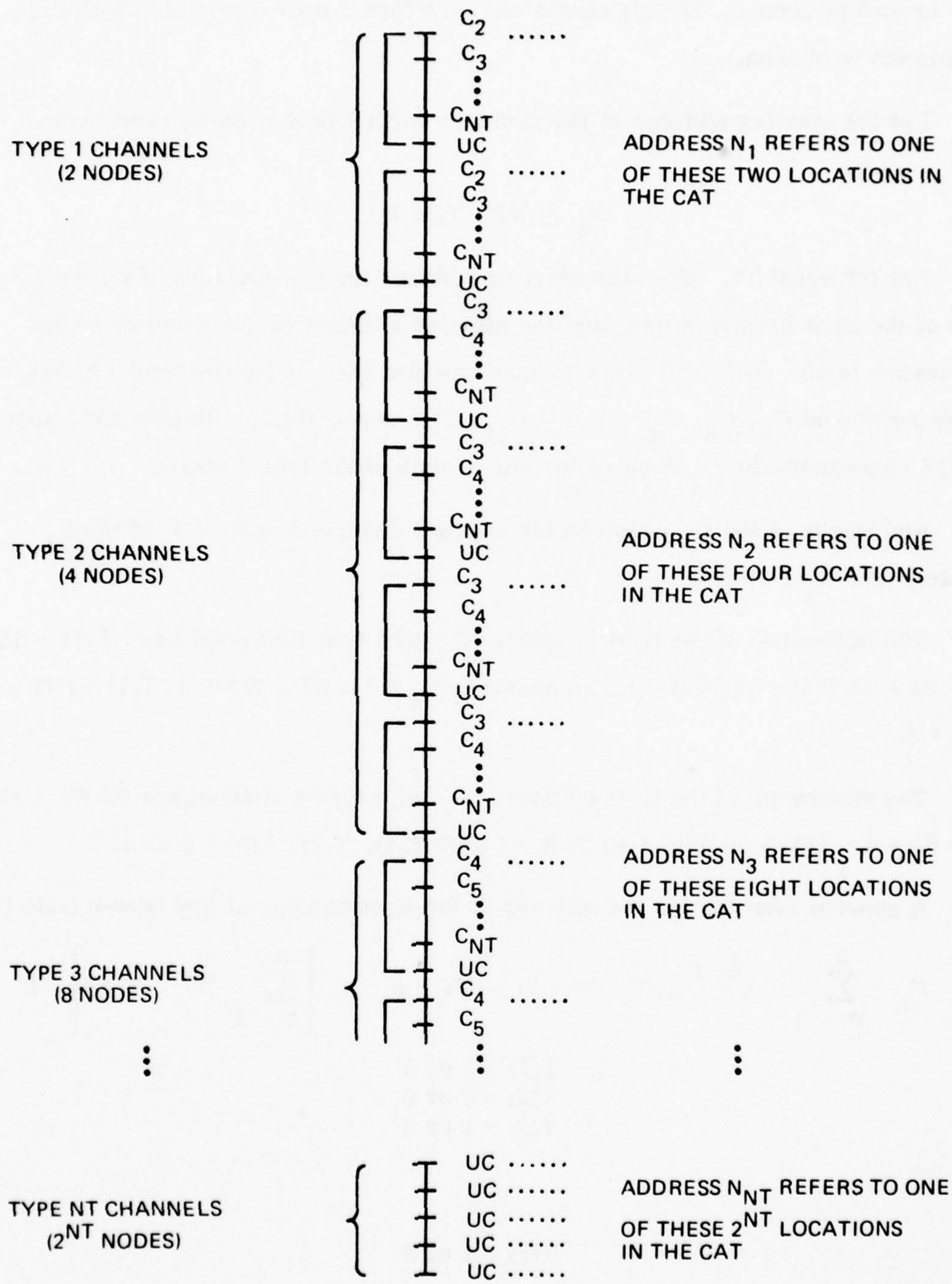


Figure 7. Illustration of How Channel Assignment Table Represents Channel Assignment Tree

zero branch is chosen. If T(1) equals one, the type 1 node associated with the one branch is chosen.

Let the starting address of the counters for a type 1 node be represented by N_1, where

$$N_1 = NT \cdot T(1) + 1$$

Let NT equal 12. Then the starting address for the counters of a type 1 node of the zero branch is one, and the starting address of the counters of the one branch is 13. This will allow 12 counters for each of the two type 1 nodes. These represent C_2, C_3, C_4 . . . C_{12}, UC, respectively. In this way, position 13 represents the C_2 counter for one branch of the type 1 nodes.

Addresses of the C_3 counters for a type 1 channel are $N_1 + 1$; of the C_4 counters are $N_1 + 2$; etc.

The addresses of the first counters (C_3) of a type 2 channel are (if NT = 12): $N_2 = 24 + 22\ T(1) + 11\ T(2) + 1$. In general, $N_2 = 2 \times NT + 2(NT-1)\ T(1) + (NT - 1)\ T(2) + 1$.

The addresses of the first counters (C_4) of a type 4 channel are (if NT = 12); $N_3 = 68 + 4 \cdot T(1) + 20\ T(2) + 10\ T(3) + 1$ with T(1), T(2), T(3) = 0 or 1.

A general formula for the address at the first counter at any type-n node is:

$$N_n = \sum_{m=2}^{n} 2^{m-1}\ (NT - m + 2) + (NT - n + 1) \left[\sum_{m=1}^{n} 2^{n-m}\ T(m)\right] + 1$$

$$\begin{aligned} T(1) &= 1 \text{ or } 0 \\ T(2) &= 1 \text{ or } 0 \\ T(3) &= 1 \text{ or } 0 \\ &\vdots \\ T(n) &= 1 \text{ or } 0 \\ &\vdots \\ T(NT) &= 1 \text{ or } 0 \end{aligned}$$

In general, the addresses of the first counters of the kth node of type n will be represented by the symbol $N_n^{(k)}$, where k equals the decimal value of T(1) T(2). . . $T_{(n)}$ plus one, for that node.

The subroutine that assigns channels in the TSSS is CATP (Reference 1). When called to make an assignment, the user's UID number and channel type are read as input parameters. The elements of the "T" array are initially set to zero. Subprogram NI is invoked to find the starting addresses for both the upper and lower nodes, starting at type 1. The counters for the requested size channel are referenced at both nodes. These counters indicate the number of assigned channels of the requested size which are contained in the channel represented by that node. Each node can contain a maximum of $2^{\text{TYPE}-i}$ channels whose channel type is "TYPE" and in which the channel type of the node being interrogated is "i". A node with a counter equal to the maximum number cannot have any channels of that size assigned within it. If the counters for both nodes contain their maximum values, no channel can be assigned, and a value of -1 is returned for the channel assignment number (NCAN). If one of the counters is less than the maximum value, that branch of the CAT tree is chosen, setting the appropriate "T(.)" number to a one or a zero. If both counters are less than the maximum value for that channel size and node position, the branch with the largest counter is chosen, in order to fill the most heavily loaded branch first. The same decision procedure is continued from left to right through the CAT tree, until a node of the requested type is reached.

When the node just prior to the requested type is reached, the user cell numbers are used to see which of the two nodes is available for assignment. If both are available, then the node associated with the zero branch is chosen.

In this way, a series of T(·) numbers are generated, which locate the horizontal path through the tree on which the assigned node is located. These T(·) numbers are then compressed into a single CAN number by invoking the subprogram CANF. This channel assignment number, along with the given channel type, specifies the node in the CAT tree which was assigned. This node

represents a specific grouping of slots in a TDMA frame.

When a node in the CAT tree is assigned, subroutine CATP will set the user's cell counter of that node to the UID number of the user requesting the channel assignment. Then subroutine ANODES is called. This subroutine updates the counters for all nodes representing channels which are contained in the assigned channel (successor nodes), all nodes representing channels which contain the assigned channel (predecessor nodes), and the assigned nodes.

2.5 MODELING THE USER COMMUNITY

2.5.1 User, Link, and Traffic Parameters

Simulation of demand assignment requires the inclusion of all factors which have an inpact on the demand for channels, namely:

1. User parameters:
   - Number of users
   - User-to-user connectivity (links)
2. Link parameters (for each link):
   - Source and destination user ID
   - Burst rate in both directions
   - Data rate
   - Full-duplex (FDX) or half-duplex (HDX) operation
   - Conference call (yes or no)
   - Number of participants in conference call and their ID
3. Traffic parameters (for each link):
   - Call rate (mean number of calls per second)
   - Mean call duration (seconds)
   - Call priority

As specified in the Statement of Work (Reference 3), the TSSS can simulate up to 2,500 users and 1,250 links. In order to define the user-to-user connectivities and the link and traffic parameters, a separate entry must be included

for each one-way link during initialization. It specifies each of the factors listed above under link parameters and traffic parameters. The call rate can be different from A to B and B to A. When a need for a call from A to B is simulated, A becomes the user requesting the call from the NCT, and vice versa. As part of this request, the user includes the number of slots needed to satisfy the call. This is computed as follows:

$$\text{Number of slots (one-way)} = \text{integer}\left(\frac{\text{data rate}}{\text{burst rate}} \cdot \text{number of slots in TDMA frame}\right)$$

For an FDX link, the number of slots needed is the sum of the number of slots needed in the two directions. For an HDX link, the number of slots needed is the greater of the number of slots needed in the two directions.

The simulation of calls at the specified rate on each link can be laborious, especially if the number of links is large. The call rate in calls per second multiplied by the frame duration yields the call rate in calls per frame. This, in turn, is equal to the probability that a call occurs on any particular frame. A straightforward but laborious simulation approach would entail at the beginning for each link simulating call occurrence probabilities. This would require the generation of a separate random number for each link of each frame. A more efficient method was described in Reference 2 and is presented in detail in Appendix A. It is the method used in the TSSS and results in Poisson distribution for the calls, with independent calls for the links at the specified call rate for each link.

After a user-to-user call is established, it is necessary to compute the duration of the call and to simulate its occurrence. It is accepted practice to assume exponential distribution for call durations, the call parameter being the mean call duration. The TSSS simulates call duration distributions by the standard method of generating random numbers with any distribution. It entails the following steps:

- During initialization, generate a table whose entries are X_{10}, X_{20} . . . X_{100}. X_i is the value not exceeded i-percent of the time in the exponential distribution of a mean value equal to 1.

- Generate a random number uniformly distributed from 0 to 1. Find the corresponding value of X_i. Use interpolation.

- Multiply the value obtained for X_i by the mean call duration. The result is the actual call duration.

Note that only a single table, which is independent of mean call duration, must be generated during initialization. It can be used for all links regardless of actual call duration.

It is desirable to simulate a mixture of calls of different priorities on any link. We can then say that priorities 1, 2 . . . 8 arrive with probabilities $p_1 . . . p_8$. When a call occurs, it is straightforward to call a random number and to use the entries $p_1 . . . p_8$ to determine the probabilities of the specific call. Since different links might have different mixtures of call probabilities, the TSSS provides for 16 such mixtures through an 8 x 16 array. An entry from 1 to 16 defines the column of eight elements to be used for a link in defining the call probability through randomization. If a link has only one priority for its calls, then the corresponding column entry is one, and the others are zero.

2.5.2 User Reactions

The behavior of the users while waiting for channel assignment or while utilizing the assigned channel is a factor influencing system performance. Various user reactions (e.g., their readiness to relinquish a channel when preempted) must be modeled. Furthermore, since these reactions are generally not known, modifying parameters which cover the range of reactions must be inserted.

In the TSSS, each user is modeled individually. He can be in any one of 15 states covering all postures from inactive through channel requests and sub-

sequent channel utilization, to channel relinquishment followed again by inactivity. In the following, the 15 states modeled in the Simulation Control (SMC) subroutine are described. While most of these states are closely related to the real-life scenario, some serve merely as a programming convenience.

State 0: user inactive. When the link for which he is the originator becomes active (through the method discussed in Paragraph 2.5.1), he enters State 1.

State 1: user activated. When the user is activated, he sends his request for a channel for the first time to the NCT.

State 2: user is in process of sending channel request. The user sends his request to the NCT the number of times dictated by the signaling strategy.

State 3: user is waiting for channel assignment. The maximum amount of time the user will wait is determined by an initialization parameter (common to all users).

State 4: requesting NCS for his position in assignment queue. The message is transmitted the number of times commensurate with the signaling strategy.

State 5: user waiting to know his position in assignment queue. If no reply arrives after a maximum wait (initialization parameter), the user gives up with a preassigned probability (initialization parameter) or requests the channel again (i.e., enters State 1). If a reply arrives and his position is less than a given value (initialization parameter) in the queue, he requests a channel again; if he is further back in the queue, he gives up with the aforementioned preassigned probability or requests a channel again (enters State 1).

State 6: shake-hands state. User received a channel assignment and is attempting to establish a link with another user. The destination user could be busy. Hence, he might be waiting in the destination user's modem queue.

State 7: waiting in modem queue. The user will wait a maximum time (initialization parameter) after which the call is canceled.

State 8: dropped call. State entered after call is canceled in State 7.

State 9: receiver of call. The user is receiving a call from another user.

State 10: check for conference call. This state is a programming convenience that permits the initialization of all participants in a conference call.

State 11: user is transmitter of call. All shake-hand operations were completed. Call is in progress.

State 12: relinquishing a channel. The duration of the call was determined through a Monte-Carlo method, as discussed in Paragraph 2.5.1. After the call is completed, a channel relinquishment message is sent to the NCT.

State 13. User is checking to see if there are any calls waiting for him in his modem queue.

State 14. Third party in conference call.

In the above description a number of initialization parameters appear which model the reactions of the users. These have a sufficiently wide range to accommodate any foreseeable user reaction. For a description of the logic of the transition between states, see Paragraph 3.4.5.

During a call the network controller may interrupt to request a disconnect and consequent relinquishment of the channel. If there is a warning of impending disconnect (delayed disconnect), the user may shorten his call and thereby complete it. If it is a request for immediate disconnect and it is not a forced disconnect system, he may relinquish with a probability which is an initialization parameter. In a forced system, he is automatically disconnected from the channel.

During a call, both users continue to check for other waiting calls with a higher priority. If there is such a call, the present call is interrupted, the user with the higher priority call becomes available, and the channel is relinquished.

2.5.3 Modem Features

The TSSS simulation also implies simulation of various modem features. The most important are:

- HDX or FDX operation
- Forced disconnect by the NCT
- Queues of variable length for call in the modem
- Multiport capability

The HDX/FDX alternatives are accommodated during initialization in the computation of time slots needed to establish a channel (see Paragraph 2.5.1).

Some modems might have a hardware feature through which the user can be automatically disconnected through the order-wire when preempted. Since the TSSS has an initialization parameter, the probability of giving up a channel when preempted, setting this probability equal to one is tantamount to a forced disconnect. The modem queue length (for incoming calls) is an initialization parameter in the TSSS.

By assigning a separate ID to each port of a modem and calling this an individual user, the performance of a multiport terminal is modeled. Multiport operation of HDX terminals is either not feasible or is unduly complex in a DA system, because some ports might want to transmit while others receive.

3. OVERVIEW OF TDMA SATCOM SYSTEM SIMULATION SOFTWARE

The simulation software has two major parts: the network control software (NCS) and the traffic simulation software (SS) (Figure 8). Basically, the NCS represents the Network Control Terminal (NCT) while the SS represents the users. The communication link between the two is an array IPBF which acts as an I/O buffer for messages between the NCT and the users.

The NCS software assigns and relinquishes channels, stores requests until channels become available, allows higher priority users precedence over lower priority users, and implements delays in transmission equivalent to specific two-way time delays. Complete flexibility of operation is provided by the NCSs acceptance of console commands at the start of simulation or during specified pauses in the processing. This provides for both batch processing and a time-sharing mode which closely resembles a real-time operating system.

The SS software contains the logic and arrays that model fictitious user terminals and maintain their status and activities on a frame-by-frame basis. It is designed for flexibility in regard to the number of users (2,500 maximum), data rates, frame rates, frame formats, slot structure, system linking scheme, priorities, and system status. After initialization, the SS activates the various links and provides the simulated communication state of all user terminals on a frame-by-frame-basis.

Messages between the NCS and the SS read into and out of the IPBF array buffer. The SS requests channels, requests position in assignment queue, relinquishes channels, and provides status information via the IPBF. The NCS assigns channels, relinquishes channels, requests retransmission, requests status, and sends position in assignment queue. Messages from the NCS are inserted. The messages to the NCS are then received and processed, and new messages sent to the users.

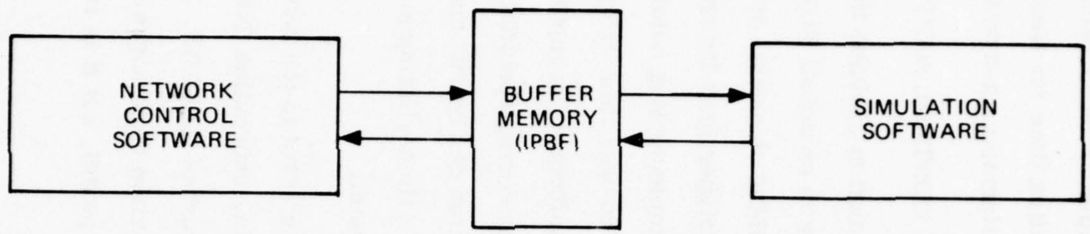


Figure 8. Network Control and Simulation Interface

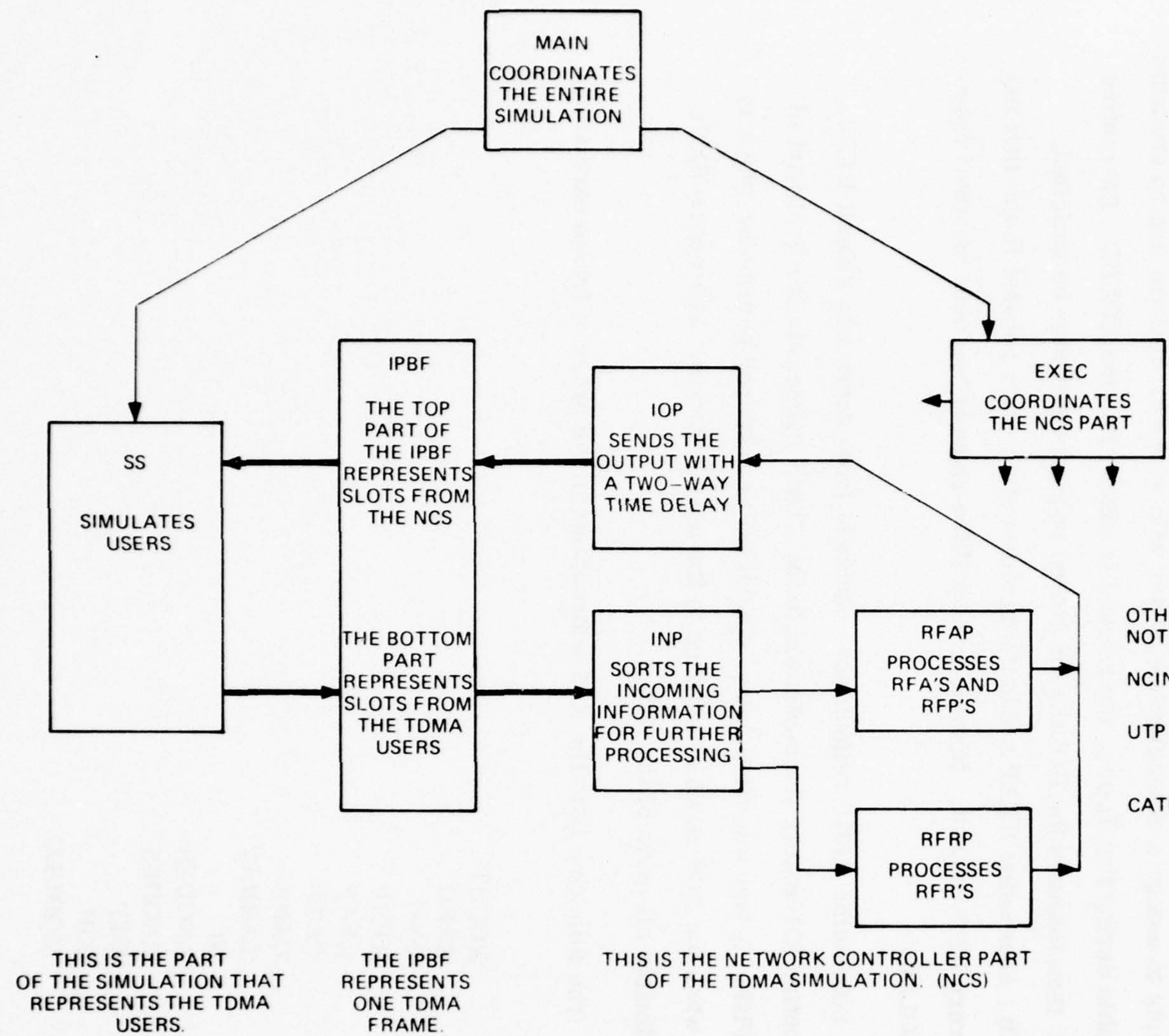
At the end of the run, a statistical summary for 12 categories correlates the NCS and SS activities during the simulation. These categories include the number of activations, completed calls, bumped calls, bumped requests from NCS, channel assignments, completed handshakes, ARQs, average waiting time for channel, average waiting time for handshake, average slot occupancy, maximum wait with no assignment, and system violations.

3.1 OVERVIEW OF THE NETWORK CONTROL SOFTWARE (NCS)

The NCS part of the TDMA simulation contains the operations normally associated with the network controller in an actual TDMA system. These duties include assigning and relinquishing channels, storing requests until channels become available, allowing higher priority users to command the use of lower priority channels, and implementing a delay in transmission equivalent to a previously specified two-way time delay. Complete flexibility of operation is provided by the NCS software's acceptance of console commands either at the start of the simulation or during specified pauses in the simulation processing. In this way, the simulation can be run unattended, as in batch processing, or it can be operated in a time-sharing mode which closely resembles a real-time operating system.

When program MAIN switches control to the subroutine EXEC (Figure 9), the NCS processing begins. First, subroutine INP is activated, which reads information from the second part of the IPBF. This information consists of various requests or status information from users. The status information is ignored in the present simulation model, but it is included to provide an extension to adaptive control.

User requests fall into one of three categories: a request for the assignment of a channel (RFA), a request for the relinquishment of a channel (RFR), or an inquiry concerning how long a previous RFA request must wait



OTHER SUBROUTINES WHICH ARE NOT PICTURED:

NCINIT – NCS INITIALIZATION SUBROUTINE

UTP – MANAGES THE HOLDQ, BUMPQ, AND USER'S TABLE

CATP – MANAGES THE CHANNEL ASSIGNMENT TABLE (CAT)

Figure 9. Overview of Network Control Software

before being processed (RFP). When the NCS part of the simulation is activated, information pertaining to user requests is read by subroutine INP. It then sorts the requests into appropriate storage queues (called RFAQ, RFRQ, and RFPQ). These requests are processed when program control shifts to subroutine RFAP or RFRP. INP checks on and drops those requests that have been stored in the HOLDQ beyond a maximum time limit. Subroutine RFAP attempts to assign a channel to each user with an RFA. If there are no channels available during that frame, the request is placed into the HOLDQ. Subroutine RFAP then inspects the HOLDQ for stored requests that may be satisfied. Finally, subroutine RFAP sends the position of a user's request in the HOLDQ for every RFP request. Nothing is sent if the request has been removed from the HOLDQ.

Subroutine RFRP relinquishes channels from users who submit RFRs. Subroutine IOP enters the messages to the TDMA users into the first part of the IPBF. A two-way time delay is modeled as a lumped parameter which is used when the NCS sends a message to the user software. Subroutine EXEC coordinates all parts of the NCS.

The following lists the NCS subroutines in the order of frame execution:

NCINIT
EXEC
INP
RFRP
RFAP
CATP
CANF
TARRAY
NI
ANODES
RNODES
UTP
IOP
MODULO

Figure 10 illustrates the relative hierarchy of these subprograms. The lowest level subroutines represent those which are called most frequently. The higher level subroutines represent those which control more activity. Lines connect the higher level subprogram with the lower level subroutines it calls. A brief summary of the primary function of each NCS subroutine follows:

NCINIT. Initializes all NCS variables and arrays before any frame processing.

EXEC. Coordinates NCS operations during each frame of a simulation run.

INP. Reads user messages from the IPBF table, and then sorts them into appropriate queues.

RFRP. Processes each user request to relinquish a channel.

RFAP. Processes each user request concerning either a channel assignment or a request for the number of users to be processed ahead of his channel request.

CATP. Performs one of three functions. In each independent section, a particular channel is assigned or relinquished, or the channel assignment table is searched for a vacant channel.

ANODES. Updates the channel assignment table for an assigned node (channel).

RNODES. Updates the channel assignment table for a relinquished node (channel).

CANF. A subprogram which computes a channel assignment number from a set of "T" numbers.

TARRAY. A subprogram which extracts the "T" numbers from a channel assignment number.

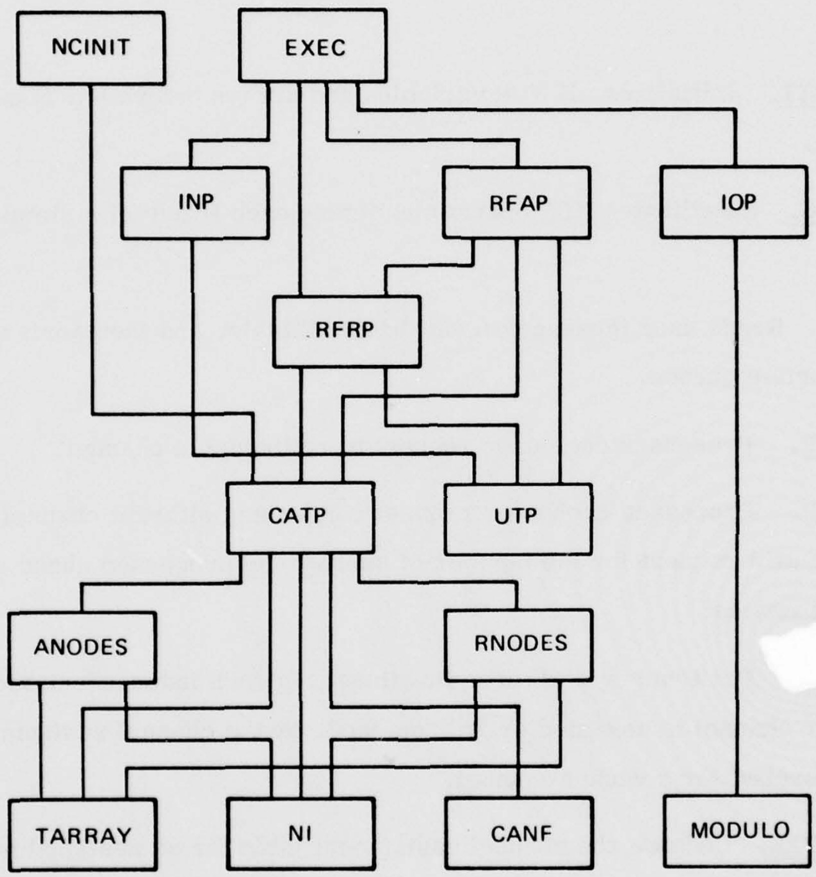


Figure 10. Overview of Hierarchy of NCS Subprograms

NI. A subprogram which computes the starting address of a node's counters in the channel assignment table.

UTP. Contains six independent sections, each of which is frequently used in other subroutines.

IOP. Writes messages from the NCS into the IPBF table. A previously specified time delay is included in IOP processing.

MODULO. A subprogram which is used to implement the time delay feature of subroutine IOP.

3.2 OVERVIEW OF SIMULATION SOFTWARE (SS)

The simulation software of the TDMA communication system is designed for flexibility in regard to the number of users (2,500 maximum), data rates, frame rates, frame formats, slot structure, system linking schemes, priorities, and system status. The following programs initialize the simulation parameters (SIMIN), simulate activation of various links (SMA), and provide the simulated communication state of all user terminals (SMC). (See Figure 11.)

The communications between the simulation and the NCS are via the IPBF buffer array. This array acts as an I/O buffer for messages between the users and the NCS. This includes channel requests, assignments, and relinquishments.

Basically, the simulation software contains the logic and arrays that model fictitious user terminals, and maintains their status and activities on a frame-by-frame basis. After initialization, the simulation cycles through three main programs for each frame, until either an end of run or a system preempt occurs.

The SMA provides the routine to activate links. Following an SMA is the main program of the simulation SMC, which controls all communications between the users and the NCS. This program first searches the IPBF buffer (via SRC) for messages from NCS, and then converts these messages into a

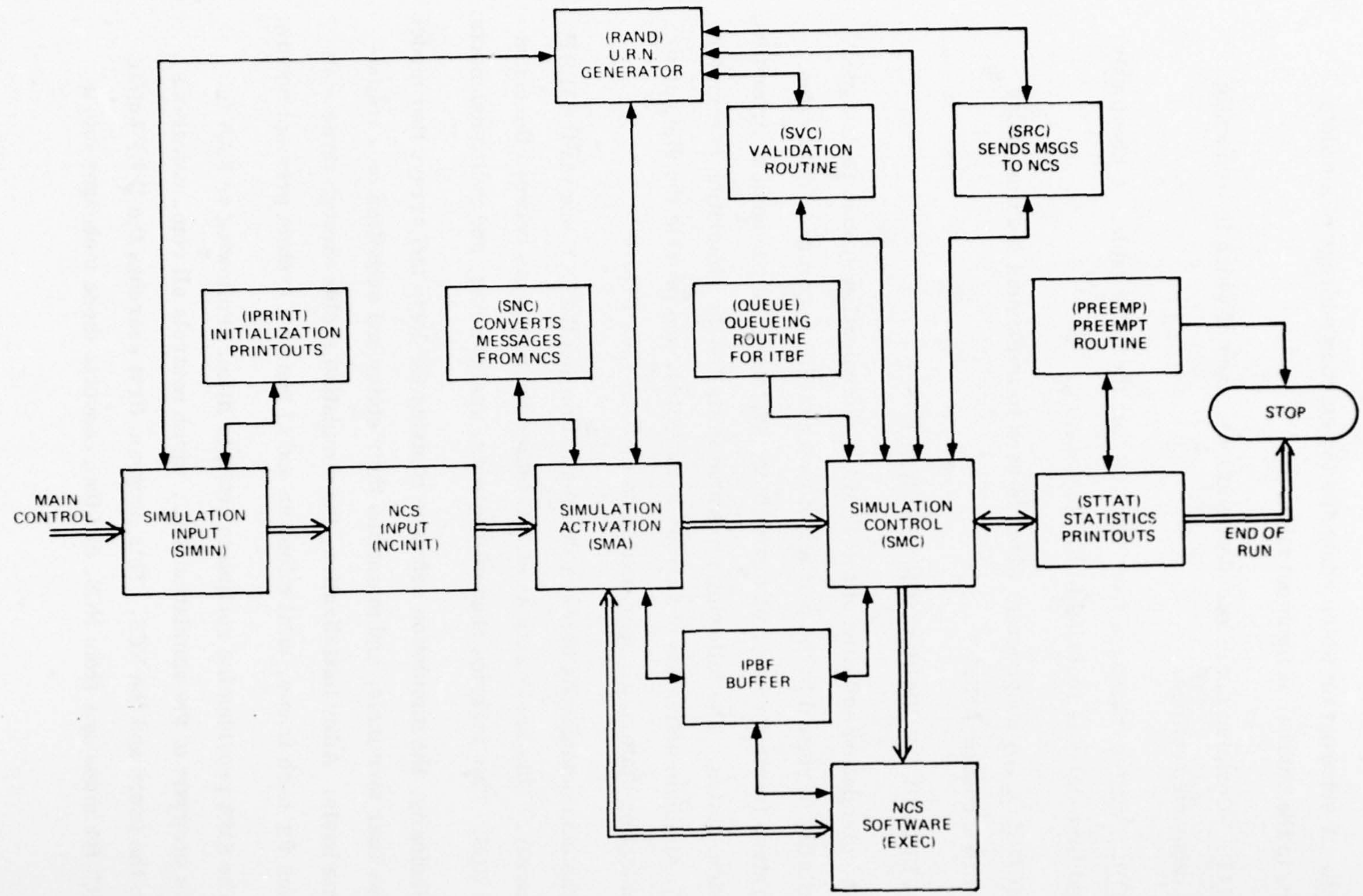


Figure 11. Overview of Simulation Software

message code which is stored in the user array ISBF. The SMC then processes each user, by first following the NCS request. If there is no NCS request, then the SMC continues according to the user's state. The SMC maintains the user's communication state, priority mode, and link information necessary to compute a call on a frame-by-frame update basis. All messages are saved in IOBF for transmission of NST times and for retransmission on ARQ requests from the NCS. To validate a call, a message must be sent NST times and received without interference at least NSS times. These counts are saved in the user array ISBF and sent with each message in IPBF.

The NCS then processes all messages from the simulation in IPBF and proceeds as explained in the NCS software section. The frame is then incremented and continues from SMA.

A subroutine (STTAT) tabulates the statistics of the simulation for 12 categories. These tables are printed at the end of the run or when requested as controlled by the IPRNT variable.

A brief summary of the primary function of each SS subroutine follows:

MAIN PROGRAM. This is the executive program that controls the flow of the subroutines. The initialization programs (SIMIN and NCINIT) are called first and only once. Then the subroutines are looped through until the number of frames exceeds the run length or until there is a system preempt.

SIMIN. Provides the input variables and values for the simulation, including those in common with the network control programs.

SMA. Is executed once per frame. The number of activations for each frame is computed and recorded. The links to be activated are then determined.

SMC. Is the main program of the simulation. It is entered once per frame and processes all devices, first by messages from the NCS and then by user state.

STTAT. Prints the statistical tables accumulated in MSTAT. There are 12 tables for 12 slot types and eight priorities.

SRC (NPP, MMX, I4). Assembles the message from the user to the NCS and sends the message ST times. The interface buffer IPBF is filled via either the user array ISBF or the IOBF.

SVC. Is used to validate entries in IPBF in the random access mode only.

SNC. Converts messages from the NCS to the user into a numbered code. The program searches the IPBF array for validated messages once per frame. A message is valid if IPBF (user, 7) equals or exceeds NSS and if IPBF (user, 5) equals NST.

QUEUE (MT, MF). Drops a user from the waiting queue of another user. It is called from various places in the SMC subroutine. Whenever a call is dropped (due to a time delay or a higher priority call), QUEUE is called.

IPRINT. Prints the initialization variables and tables as produced in subroutine SIMIN. If INIT is 0, this subroutine is not entered.

PREEMP. A small subroutine which is called only if there is a system preempt. It prints the frame number at which the preempt occurred and then calls the statistics subroutine STTAT. PREEMP then stops the run.

RAND (I, R). Provides the uniform random number by calling the system uniform random number generator program random-$uniform (i,r). The system subroutine is in PL1, so RAND is needed to interface the FORTRAN subroutines with the PL1 system subroutine.

3.3 DETAILED DESCRIPTION OF NCS

3.3.1 Description of NCS Data Files and Arrays

This section describes the structure and implementation of NCS data arrays (Figure 12). Because of the large size of many of these arrays, they require special attention. Network control requires that the status of each user in the NCS be maintained to handle any user request. The NCS must know if a user already has a channel assigned to it as well as the priority, channel size, and particular grouping of slots which are assigned. In addition, the NCS must know if a user's request for a channel assignment has been waiting for processing from a previous frame. A user with a low priority may be forced to relinquish his channel if a user with a high priority requests a channel. These functions are performed by the use of three interrelated tables: holding queue (HOLDQ), bumping queue (BUMPQ), and user's table (UT).

User's Table (UT)

The UT contains the current status of each user in the NCS, for up to 2,500 users. Each user has an integer user's identification number (UID) between 1 and 2,500. The UID can be used to refer to a particular row in a table containing user status information in the NCS. The UT is arranged with 2,500 rows, each of which contains six elements. These elements contain the user's priority, PR(·); channel type, TYP(·); channel assignment number, CAN(·); successor number, S(·); predecessor number, P(·); and timing or clock element, CL(·).

The user's priority consists of an integer between 1 and 8, which represents the command authority of the user. The higher the priority number, the greater the authority. In this way, a priority 6 user may command the use of a priority 3 user's channel, if it becomes necessary.

The channel type number consists of an integer between 1 and the variable NT. NT represents the total number of different size channels. Channels

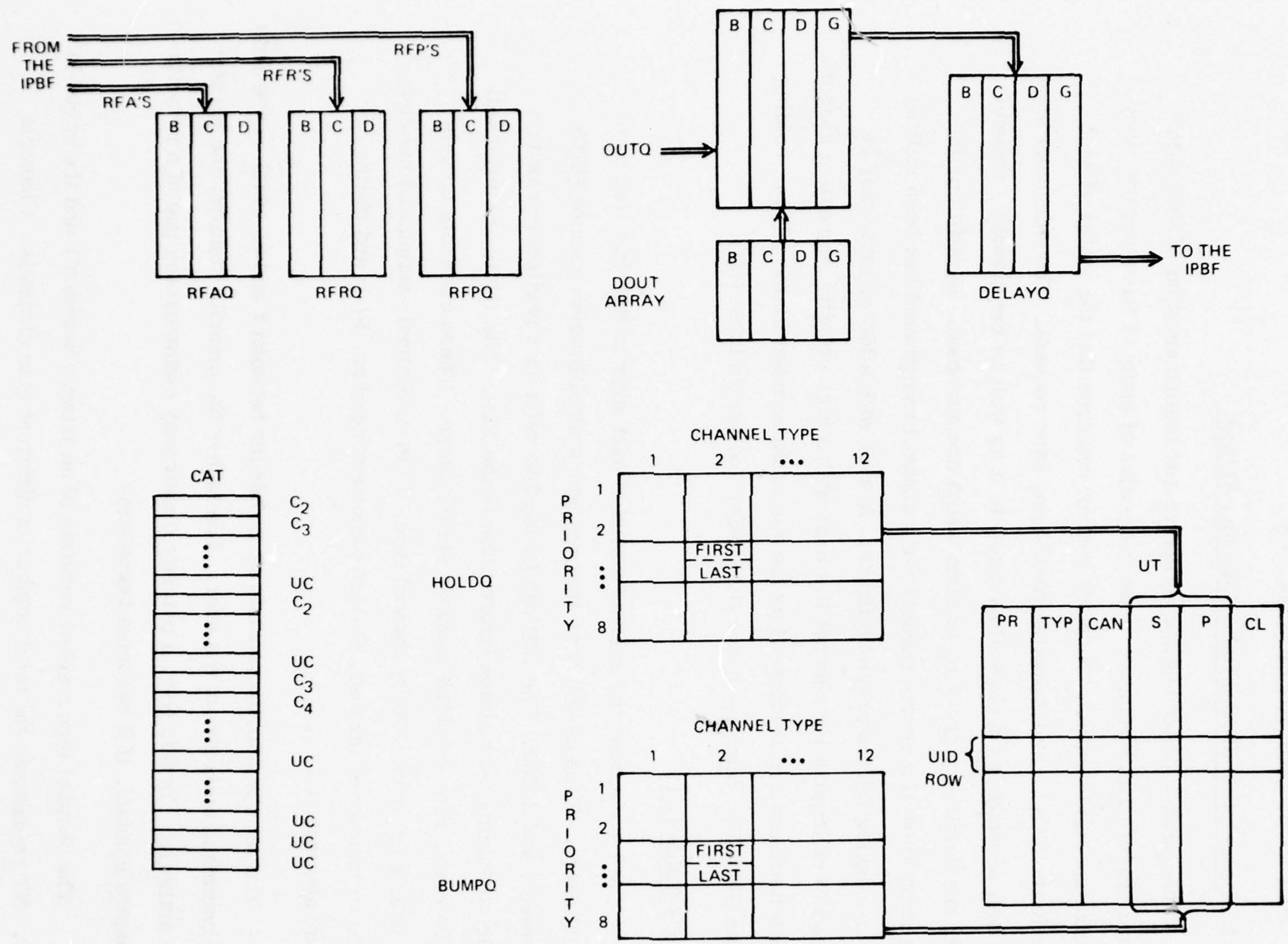


Figure 12. Overview of Major NCS Arrays

containing more slots are represented by lower channel type numbers. The initial postulate is that a user will request only a number of slots equal to some power of two. This assumption is necessary to keep the NCS model from becoming too complex to simulate. The largest channel contains half of the total available slots in a frame. All the slots in the frame cannot be assigned as a channel, because some slots are used for overhead purposes. The channel type number can be converted to the channel size in number of slots using the following equation:

$$\text{slots} = 2^{NT - \text{channel type}}$$

$$\text{channel type} = 1, 2 \ \ldots \ NT$$

The channel assignment number is an integer quantity which can take on any value between 1 and 2^{NT}, if a channel is assigned to this user, or a -1 otherwise. This CAN number, with the channel type, specifies an exact grouping of slots in a frame.

The successor and predecessor entries contain the UID numbers of a user which precedes the given user and a user which succeeds the given user in one of two tables. The same UT entries are used to specify a user's position in either the HOLDQ table or the BUMPQ table. This is possible because HOLDQ and BUMPQ entries are mutually exclusive.

The CL entry is an integer quantity which takes on the values 0 ... MAXC, if that user resides in the HOLDQ table, or a -1 otherwise. CL represents the number of frames the user's RFA request has been waiting in the HOLDQ table. If the value of CL reaches MAXC (a preassigned constant), then the user's request is removed from the HOLDQ table and CL is set to -1.

Holding Queue (HOLDQ)

The HOLDQ contains the RFA requests from users which were not able to obtain channel assignments during previous frames. This table is divided

into 96 independent queues that are ranked according to channel type and user's priority. Since each queue could possibly contain 2,500 users, the size of this table could be tremendous. The UT is used to reduce the HOLDQ to a reasonable size. Each HOLDQ presently contains two elements. These two entries contain the UID number of the first user in that queue, and the UID number of the last user in that queue.

In the UT, the successor of the first HOLDQ entry gives the UID number of the next successive entry in that HOLDQ. The successor of that UID number gives the UID number of the next successive entry in that queue. This will continue until the UID number of the last user in that queue is reached. In a similar way, the predecessor of the UID number of the last queue entry will give the UID number of the next-to-last queue entry. Its precedessor gives the UID number of the second-from-last entry of the queue. This will continue until the UID number of the first queue entry is reached. In this way, all HOLDQ entries can be referred to by the use of 2,500 sets of successor and predecessor numbers.

Bumping Queue (BUMPQ)

The BUMPQ table contains the UID numbers of every user which has been assigned a channel. The table is grouped by user priority and channel type. The table is referred to by a user who is searching for a user of lower priority in order to commandeer its channel. This procedure of searching and relinquishing is called bumping a user from a channel. The operation of the BUMPQ table is identical to that of the HOLDQ table. The same successor and predecessor numbers in the UT are used for both tables. This is possible since all users who are assigned channels are members of the BUMPQ and since all HOLDQ members do not have channels assigned them. In this way, the channel assignment number for each user can determine if that user is in the HOLDQ table or the BUMPQ table.

Channel Assignment Table (CAT)

In addition to storing the status of each user in the NCS, the status of every slot in the present frame must be considered. This is necessary if the control system is to respond correctly to channel assignment requests. The CAT stores information in the form of a "channel assignment tree", which is discussed further in Paragraph 3.3.7. The CAT stores all pertinent information regarding the state of the slots in the present TDMA frame.

Frame Processing Storage (RFAQ, RFRQ, RFPQ)

When messages from users are read out of the IPBF table, they must be sorted with regard to the purposes of the requests. RFAs are placed into the RFAQ table, RFRs are placed into the RFRQ table, and RFPs are placed into the RFPQ table. The B, C, and D parts of an IPBF entry contain the UID number, priority, and type of the user request. The G part determines what kind of a request is being submitted. The value of the G part of an IPBF entry determines into which of the three queues (or none) the B, C, and D information is to go.

Output Queue (OUTQ)

When a section of the NCS processing has a message to send to a user, it writes the UID number of the user who is to receive the message, and three other numbers which carry information. These are to be written into the B, C, D, and G parts of the IPBF, and repeated NST times. Consequently, the OUTQ table stores one copy of the B, C, D, and G numbers of an output message.

Delayed Output Queue (DELAYQ)

The DELAYQ table introduces a specified time delay (expressed in the number of slots) between reading an OUTQ entry and writing a message into

the IPBF. This time delay simulates the NCS-to-user round-trip propagation delay. The next OUTQ entry for processing is entered into the first position of the DELAYQ table. The elements of the DELAYQ are moved down one position, with the last DELAYQ element being lost. The last DELAYQ entry is then written into the IPBF and repeated NST times. A new OUTQ entry is entered into the top of the DELAYQ table, and the process is repeated until either the OUTQ is empty or the NCS portion of the IPBF is filled for the present frame. In this way, the length of the DELAYQ table determines how many NCS linking slots the output message will be delayed during its transmission. This delay feature is implemented in a different way to save computation time. The description of subroutine IOP (Paragraph 3.3.14) explains this in greater detail.

Delayed Output Array (DOUT)

When the NCS wants a user to relinquish a channel, it sends a message to that user. If there is a time delay before the user must sign off (tau greater than 0), then the NCS sends two messages. One message is sent immediately, which instructs the user to prepare for relinquishment of the channel in tau seconds. The second message is sent at the end of the allotted time, which instructs the user to relinquish his channel immediately. This message is placed into the DOUT array at the same time the initial message is placed into the OUTQ table. The DOUT array places the delayed message into the OUTQ table at the end of the specified time. The ARRAY table determines which users have already been requested to relinquish their channels, in order to avoid sending duplicate messages to the same user.

The TWOEXP (12) array is initialized to contain powers of two, ranging from 2^1 to 2^{12}. This table is later referenced by the NCS subroutines. The use of the TWOEXP array saves computation time by eliminating many exponential operations and multiplications.

3.3.2 Subroutine NCINIT (NCS Input and Initialization)

This subroutine initializes the arrays and variables which were not initialized in the user simulation. This procedure includes reading information from the terminal, which can be printed out in summary fashion, if desired. Future console commands may be entered at the beginning of a simulation run. These commands are executed during the frame which is specified with the command information. Additionally, slots may be permanently allocated, the status of each user may be requested periodically, and the operator may choose to enter the control of the program during future frames. The latter option is ideally suited for the execution of the simulation in the time-sharing mode to simulate a real-time environment.

3.3.3 Subroutine EXEC (NCS Executive Control)

This subroutine coordinates the NCS part of the simulation. During the execution of one TDMA frame, the main program calls EXEC, which then calls INP, RFRP, RFAP, and IOP.

Incoming messages are read in from the top portion of the IPBF and sorted according to whether they contain RFA, RFR, RFP, or user status information. The first three message types are placed into either RFAQ, RFRQ, or RFPQ. The user status information is ignored in the present simulation model. User requests which have been waiting for a channel assignment are located in the HOLDQ. If a HOLDQ entry has waited a specified number of frames and still has no channel assigned, it is dropped from the HOLDQ.

Users who have been requested to relinquish a channel are processed next. This is done before the channel assignment requests are considered, in order to leave more vacant channel space. Channel assignments are then made. Any request which cannot be immediately processed is placed into the HOLDQ, with the possibility of being assigned to a channel in a future frame.

The HOLDQ is next compared with available slots, since there may be some slcts left in the frame which can satisfy some previous channel assignment requests. Now the HOLDQ has been filled for the present frame.

Requests for a user's position in the HOLDQ (RFPs) are now processed. A check is made to see if all available slots in this frame are taken (system saturation). Appropriate information is given to the user's part of the simulation. Messages for transmission to the users are stored in the output queue (OUTQ). These are placed into the second half of the IPBF, with a time delay, expressed in slots, being taken into account. This time delay is the representation of a two-way time delay in a TDMA system.

Information is read in from the terminal, if desired. This enables the operator to request the status of a user, to request the status of a particular channel, or to enter console commands for execution during the next frame. This frame-to-frame control gives the simulation the flexibility of operating according to an adaptive experimental procedure, as well as providing a suitable model for console control of the system.

When the NCS processing is interrupted to input data between frames, the first query printed is "Do you wish to inquire about a user? (1=y, 0=n)." After a positive response, the next query requests the UID number of the user which is to be investigated. Various answers by the simulation may be given, depending on whether the user has a channel assignment or has none, and if a user is waiting for a channel assignment or not. The next section of input prints the query, "Do you wish to inquire about a particular channel? (1=y, 0=n)." The affirmative response will request the input of a channel type and a channel assignment number. The third section of of input reads console commands from the terminal, which are to be immediately processed by the NCS. This starts with the query, "Do you wish to enter a console command? (1=y, 0-n)." The affirmative response will prompt the NCS to request the entry of specific information as the user's ID number, channel type, user's priority and the kind of command this is to be.

3.3.4 Subroutine INP (Input Message Sorting)

This subroutine reads user messages, which are sent to the NCS, from the top portion of the IPBF. These are sorted into RFAQ, RFRQ, or RFPQ according to the content of the messages. Console command information from the CPDATA (100,5) array is read like the IPBF data when NFR equals the given frame number. Next, the time counters (CL) of each HOLDQ member is incremented by 1 to indicate that an additional frame of waiting time has passed. Users whose counters exceed the maximum number of frames (MAXC) are dropped from the HOLDQ table.

3.3.5 Subroutine RFRP (Channel Relinquishment Processing)

This subroutine relinquishes channels from users according to the information stored in the RFRQ table. The RFRQ contains the UID number of each user, the priority, and the type of channel which is to be relinquished. A check is made to see if a channel is assigned to this user. If no channel is assigned, the request for channel relinquishment is ignored. In this way, repeated receptions of the same message will have no detrimental effect on NCS operation.

Each entry in the user's table (UT) has an associated CAN entry (channel) assignment number). If user UID has no channel assigned to it, CAN (UID) will be -1. If user UID has a channel assigned to it, CAN (UID) will be some positive number between 0 and 4,096, depending on the number of channel types (NT), type of channel, and location of the channel in the frame. The CAN number will be explained further in Paragraph 3.3.7. The specified channel is removed from the channel assignment table and from the BUMPQ table. The user's ARRAY entry is then set to 0 to enable a new user assignment to receive a bump request message. The CAT records what particular slots have been occupied by a user's channel. The BUMPQ array records what assigned channels may be preempted by higher priority channels. Since this table locates

users who may be "bumped" to make room for higher priority users, it is called the BUMPQ. The BUMPQ is arranged as the HOLDQ. Finally, changes are made in the UT to reflect that user UID has no more channels. This includes setting CAN (UID) to -1 and setting the successor and predecessor numbers (for the BUMPQ in this case) to 0. This procedure is continued until all members of the RFRQ have been processed.

3.3.6 Subroutine RFAP (Channel Assignments)

This subroutine performs three major tasks. First, current members of the HOLDQ table are compared with the channel assignment table in an attempt to satisfy some old channel assignment requests. Next, it processes each member of the RFAQ (requests for channel assignments), placing a request into the HOLDQ table if the requested channel is not available. If a channel is already assigned to the user, it is checked to see if it is of the same size. If not, it is relinquished and a new channel is sought. If it is the same size, a new message is sent while the NCS keeps the channel assignment. Third, each member of the (RFRQ) is processed to find the number of channel assignment requests ahead of the user in the HOLDQ. Only channels of the same or smaller size are considered in computing the position of a user in the HOLDQ.

3.3.7 Subroutine CATP (Channel Assignment Table Processing)

Subroutine CATP performs one of three functions, depending on the value of variable CATSW. In each of the three independent sections, the channel assignment table (CAT) is updated to reflect the change in status of the TDMA slots. The three functions are:

(CATSW = 1)	To search the CAT table for a channel of a particular size, and to assign it to a given user.
(CATSW = 2)	To assign a specific channel to a particular user.
(CATSW = 3)	To relinquish a particular channel from a user.

3.3.8 Subroutine ANODES (Assigned Node Updating)

Subroutine ANODES (NII, CAN, TYPE) will update all predecessor and successor nodes in the CAT table for an assigned node (which represents a channel). Subroutine ANODES is composed of two sections. The first section updates the counters for all predecessor nodes and the assigned node. The second section updates the counters that are associated with all successor nodes.

3.3.9 Subroutine RNODES (Relinquished Node Updating)

Subroutine RNODES (NII, CAN, TYPE) will update all predecessor and successor nodes in the CAT table for a relinquished node (which represents a channel). Subroutine RNODES is composed of two sections. The first section updates the counters for all predecessor nodes and the assigned node. The second section updates the counters which are associated with all successor nodes.

3.3.10 Function CANF (Channel Assignment Number Computation)

Function CANF (TYPE) is a subprogram which computes the channel assignment number (CAN) for a channel that is specified by a set of T(·) numbers and the type (TYPE) number. A CAN number with a channel type number will uniquely specify a grouping of slots in the TDMA frame (a channel).

The purpose of the channel assignment number is to specify an exact TDMA channel by the use of a single number (not including the channel type number). This is accomplished by letting the 12 T(·) numbers be represented by the 12 least significant bits of an integer quantity, called the CAN number.

3.3.11 Subroutine TARRAY (Computation of T (·) Numbers)

Subroutine TARRAY (CAN, TYPE) is a subprogram which converts a given channel assignment number (CAN) and a given channel type (TYPE) into a series of T(.) numbers. These T(·) numbers can be used to trace a path through the channel assignment tree.

3.3.12 Function NI (Starting Address for Particular Node in CAT Table)

Function NI (i) is a subprogram which computes the starting address of counters for a type-i node which is located on a path in the channel assignment tree. This path is uniquely specified by the T(·) array numbers.

3.3.13 Subroutine UTP (Manipulation of HOLDQ, BUMPQ, and Users Tables)

Subroutine UTP (UID, PRI, TYPE) contains six independent sections, each of which performs a frequently used function. These sections control most changes to the HOLDQ table, the BUMPQ table, and the user's table. In order to access one of the sections, the integer variable UTSW is assigned a value between 1 and 6 prior to calling UTP. The function of each section is now described.

UTSW = 1	This section places user UID at the end of one part of the HOLDQ table, as specified by the user's priorty (PRI) and channel type (TYPE).
UTSW = 2	This section removes user UID from its part of the HOLDQ table that is specified by the user's priority (PRI) and channel type (TYPE).
UTSW = 3	This section places user UID at the end of one part of the BUMPQ table, as specified by the user's priority (PRI) and channel type (TYPE).
UTSW = 4	This section removes user UID from its part of the BUMPQ table, as specified by the user's priority (PRI) and channel type (TYPE).

UTSW = 5	This section requests the number of users which presently occupy the HOLDQ, as specified by user's priority (PRI) and channel type (TYPE). The information is returned to the calling subroutine through variable UID.
UTSW = 6	This section searches the BUMPQ table for a bumping candidate that will release at least one channel of type (TYPE) and priority (PRI) when requested. If a user was found that satisifies these conditions, its user identification number is returned through variable UID. If no suitable user was found, variable UID is set to -1.
UTSW = 7	Continue searching for a bumping candidate, starting at user UID in the BUMPQ table.

3.3.14 Subroutine IOP (Output Message Processing)

This subroutine writes the output messages into the IPBF table. A previously specified two-way time delay is modeled with the table DELAYQ. Delayed messages are transmitted NST times, with the ith part of the IPBF entry incremented by 1 if a random number is larger than a specified probability. Delayed requests for users to relinquish their channels (for tau greater than 0) are implemented by a delayed output message table (DOUT).

3.3.15 Function MODULO (MOD 3000 Number Converter)

Function MODULO (ARG) is a subprogram that changes a given integer, ARG, into its corresponding integer which lies between 0 and 2,999. This function is used to compute addresses in the delayed output table (DELAYQ). This table models a lumped two-way time delay and is contained in subroutine IOP.

3.4 DETAILED DESCRIPTION OF SS

3.4.1 Description of SS Data Files and Arrays

This section explains the file and array structures of the main variables of the simulation software (Figure 13).

Input Data File (IDF). For each link, a data line on file is read into the computer and then connected to INPT and XINPT entries. Since the file is less than 80 characters per line, a card file could also be used.

MMP - I10 - From (P)

MMQ - I10 - To (Q)

AL - F10.2 - Mean message length in seconds

XRS - F10.2 - Activation rate in calls per second

IPG - I5 - Priority group 1 to 16

BRPQ - F10.2 - Burst rate from P to Q

BRQP - F10.2 - Burst rate from Q to P

DATAR - F10.2 - Data rate

LTHF - I2 - Link type; HDX = 2, FDX = 1

NCONF - I2 - Number of conference calls

If NCONF = 0, there is no conference call and the next link is read in.

If NCONF = MM, then another data line is read in for MM user IDS.

IDC (N), N = 1, MM with format of (MM) I10.

IDC entries are then stored in array LCONF.

IDC and LCONF (N) - Conference Call Arrays

IDC - Input array for conference calls. Used only in SIMIN.

LCONF - Array to store conference calls per link. If a link has a conference call, the address of N of the number of others in the call is given and followed by the user number of those in call.

0–10	10–20	20–30	30–40	40–45	45–55	55–65	65–75	75–77	77–79	79–80
MMP (TO) I 10 INTEGER	MMQ (FROM) I 10 INTEGER	AL (MESSAGE LENGTH) F 10.2 DECIMAL	XRS (ACTIVATION FACTOR) F 10.2 DECIMAL	FPG (PRIORITY GROUP) I 5 (1 TO 16) INTEGER	BRPQ (BURST RATE) MMP MMQ F 10.2 DECIMAL	BRPQ (BURST RATE) MMP MMQ F 10.2 DECIMAL	DATAR (DATA RATE) F 10.2 DECIMAL	LTHF 1–FDX 2–HDX	NCONF I 2 H	

IF NCONF ≠ 0, THEN NCONF = NUM
READ IN NEXT RECORD

IDC (M), M = 1 TO NUM

0–10	10–20	20–30	30–40	40–50	50–80
IDC (1) I 10 (USERID)	IDC (2) I 10 (USERID)	IDC (3) I 10 (USERID)	IDC (4) I 10 (USERID)	IDC (5) I 10 (USERID)	

Figure 13. Input Structure for Data File IDF

For example, INPT (X, 5) = 24

Link X has a conference call. M = LCONF(24), where M is the number of users in call.

ID = LCONF(24+J), J = 1, M; where ID is the user IDS of these conferences.

INPT (NL, 6) and XINPT (NL) - Input Link Arrays. From the data file IDF the working arrays INPT and XINPT are computed to contain all information needed for a link. NL is the number of links.

INPT (*, 1) - From

INPT (*, 2) - To

INPT (*, 3) - Priority group from 1 to 16

INPT (*, 4) - Slots per link

INPT (*, 5) - Conference call address in LCONF or 0

INPT (*, 6) - Mean message length in frames

XINPT (*) - RS - Accumulative calls per frame

ISBF (2500, 10) - User Array. This array maintains the user status and current call information on a frame-by-frame update basis.

ISBF (*, 1) - Validation count (I of IPBF)

ISBF (*, 2) - MWHO - Other user of the link (to or from)

ISBF (*, 3) - Priority of call

ISBF (*, 4) - State of user - 0 to 14

ISBF (*, 5) - Slots per link as power of 2 (the N of 2^N)

ISBF (*, 6) - Conference call 0 - no conference call
XX - array location of LCONF as explained above

ISBF (*, 7) - Linking slot locations and how many per frame. The first part of the number is the first location and the last two digits are how many per frame; i.e., 30203 is location 302 in IPBF and three consecutive slots per frame.

ISBF (*, 8) - Message count (F of IPBF) up to NST

ISBF (*, 9) - Message code from NCS messages

0 - None
1 - Assignment
2 - Disconnect - no delay
3 - Disconnect - tau delay
4 - ARQ
6 - Request status
-XX - Position in assignment queue

ISBF (*, 10) - Mean message length

IOBF (2500, 4) - Last Message Buffer. In order to send a message NST times or answer an ARQ, this information is saved from IPBF.

IOBF (*, 1) = IPBF (*, 1) - User ID

IOBF (*, 2) = IPBF (*, 2) - Priority

IOBF (*, 3) = IPBF (*, 3) - Slots per link or user status

IOBF (*, 4) = IPBF (*, 4) - Message

XPRT (16, 8) - Priority Table. XPRT (L, M) is a representation where L is from 1 to 16 for the priority groups and M is from 1 to 8 for the priorities. The lowest priority is 1, and the highest priority is 8.

This table is read in either from a file or from the console, with the probability of each group having each of the eight priorities. These values are cumulative, with 1 having a value of 1.0. For example:

Input for Group 5	XPRT after Summation
XPRT (5, 1) = .1	XPRT (5, 1) = 1.0
XPRT (5, 2) = .2	XPRT (5, 2) = 0.9
XPRT (5, 3) = .1	XPRT (5, 3) = 0.7
XPRT (5, 4) = .1	XPRT (5, 4) = 0.6
XPRT (5, 5) = .1	XPRT (5, 5) = 0.5
XPRT (5, 6) = .1	XPRT (5, 6) = 0.4
XPRT (5, 7) = .1	XPRT (5, 7) = 0.3
XPRT (5, 8) = .2	XPRT (5, 8) = 0.2

LP (8) - Linking Slots for Random Access. For the random access mode, the number of 2^n slots for each of the eight priorities is LP (J) where J is 1.8 and LP (J) is n.

ITIM (2500) - Time Count Buffer. Time frame buffer for each user in frames.

RS (301) - Cumulative Poisson Activation Probability. Storage table created by SIMIN for the cumulative addition of the Poisson activation probability for 0 to 300 link activations per frame. It is used to obtain the number of links for activation during a frame.

RXL (11) - X/LAMDA Table. Table used to reduce the actual simulated message length computation in SMC. It is based on the exponential probability density function for a random variable X.

$$F(X) = \frac{1}{\lambda} e^{-\frac{X}{\lambda}}$$

Interpolation is used for estimates.

IPBF (4097, 7) - Linkage Buffer between NCS and User or Simulation. Below is an explanation of input to IPBF from the user as controlled by the simulation.

In mode LLX = 1 (demand access) and LLX = 2 (polling), each slot is preassigned as designated by ISBF (*, 7). In mode LLX = 3 (random access), the linking slots are assigned by priority as in LP.

IPBF (*, 1) - B - From the user ID

IPBF (*, 2) - C - Priority of call as in ISBF (*, 3)

IPBF (*, 3) - D - Slots per link or user status

IPBF (*,4) - G - Messages:

1 - Request for assignment
2 - Relinquishment of channel
3 - Request position in assign queue
4 - User status

IPBF (*,5) - F - Message count from 0 to NST

IPBF (*,6) - H - Interference in random access mode indicator by incrementing for each entry

IPBF (*,7) - I - Counter for validated messages, corresponds to ISBF (*,1)

Note that a message is only acted upon if F = NST and 1 NSS.

PEBF (2500,2) - Probability of Message Interference

PEBF (*,1) - Probability of message interference from user to NCS.

PEBF (*,2) - Probability of message interference from NCS to user.

ITBF (2500,6,2) - Waiting Queue and Priorities. For each user there is a waiting queue up to MQ with 6 as the maximum. Each user in the queue has an associated priority of the call which is also stored.

ITBF (*,Q,1) - User ID

ITBF (*,Q,2) - Priority of call

Each queue is ordered according to the priorities of the callers. As new calls are added, the ITBF array is expanded up to MQ. The lowest priority call is dropped if the queue length is exceeded.

MSTAT (12,12,8) - Statistics Table. This array contains all the statistical information. There are 12 categories, and the tables are printed as a matrix for slot type (1 to 12) and priorities (1 to 8). Totals are given for each column and row.

MSTAT (1,*,*) - Activations

MSTAT (2,*,*) - Completed calls

MSTAT (3,*,*) - Bumped calls

MSTAT (4,*,*) - Bumped requests from NCS

MSTAT (5,*,*) - Channel assignments

MSTAT (6,*,*) - Completed handshakes

MSTAT (7,*,*) - ARQ

MSTAT (8,*,*) - Average waiting time for channel

MSTAT (9,*,*) - Average waiting time for handshake

MSTAT (10,*,*) - Average slot occupancy

MSTAT (11,*,*) - Maximum wait/no assignment

MSTAT (12,*,*) - System violations

3.4.2 Main Program

This is the executive program, and it controls the flow of the subroutines. The initialization programs (SIMIN and NCINIT) are called first and only once. Then the subroutines are looped through until the number of frames exceeds the run length, unless there is a system preempt.

The Simulation Activation (SMA) is called and then followed by the Simulation Control (SMC). When the simulation programs are finished, an output message giving the state of the users may be printed (IPRNT is 1). Intermediate statistics may be requested too.

Control is then given to the NCS subexecutive program EXEC, which runs the NCS subroutines. When the NCS returns to MAIN, the output buffer IPBF may be printed, depending upon the value of IPRNT. MAIN then loops, starting from SMA and lasting until the end of the run. The statistics are then printed in 12 tables by subroutine STTAT.

3.4.3 Subroutine SIMIN (Simulation Input and Initialization)

The simulation input and initialization routine provides the input variables and values for the simulation, including those in common with the network control programs.

The first record is from the console. This input defines the file numbers of the initialization files and of the output files. If a file number is 5, that input is via the console. If a file number is 6, the output is via the console.

All arrays are initialized to 0. Other variables are also initialized, including the number of frames set equal to 0.

To facilitate a terminal input, first a statement of the requested variable is printed and then the value is typed from the console. The list of these variables is described in the User's Guide.

Many variables are entered in seconds. These are then converted to frames relative to the frame rate.

The data file for arrays INPT and XINPT contains the simulation linking structure. Each record represents a link. Each user-to-user link is considered to be a one-way link; i.e., P to Q and Q to P are two independent links. Each record of the linkage file contains the origin and distribution (P and Q), the mean message length, the average activation rate, the priority group of the user, the burst rate from P to Q, the burst rate from Q to P, the data rate, the indicator for full-duplex or half-duplex, and the number in a conference call.

To determine the number of slots per link for each link, the following computations are made:

CE = Data rate * slots per frame

F = CE/Burst rate from P to Q

G = CE/Burst rate from Q to P

For full-duplex, the number of slots needed to establish a link is F + G. For half-duplex, the number of slots is the larger of F and G.

Since the linkage structure is based upon powers of 2 up to 2^n, where n equals 12, the number of slots per link is converted to a power of 2 by use of table ITWO. This n is then stored in INPT (I, 4).

Conference calls are stored in array LCONF. If there is no conference call for a particular link, INPT (I, 5) is 0. Otherwise, if INPT (I, 5) is 10, then 10 is the location in LCONF where the number of participants for that call is followed by the identities of the parties. For example, if

LCONF (10) = 3

LCONF (11) = 1

LCONF (12) = 7

LCONF (13) = 15

then there are three participants who are users 1, 7, and 15. This method saves storage by condensing the array sizes, since not all links have conference calls and only a few links may have the maximum number of participants.

The cumulative activation rate x_i^1 for every link i is computed as follows: If x_i^1 is the average activation rate of link i, for i = 1, 2, 3 . . . N expressed in minutes per activation $(act/min)^{-1}$, then

$$x_i^1 = x_i + x_{i-1}^1$$

If x_i equals 1, there is one activation per second for link i.

The cumulative activation rate computed for every link replaces the average activation rate that was the original input XINPT.

The linking slot assignment is designated by a file IE and stored in ISBF (I, 7) for a mode LLX of 1 or 2. The structure is XXXXYY, where XXXX is the

slot location in IPBF buffer array and YY is the number of slots. In the random access mode, an array LP defines the number of linking slots for each priority. Only the number of slots needed is stored in ISBF (I, 7), or YY as above.

The cumulative Poisson activation probability table RS is computed for up to 88 activations per frame using the Poisson probability equation.

$$P_k = \frac{\lambda_\tau^k}{k\,!} \cdot e^{-\lambda_\tau}$$

where k is the number of activations per frame (88), and $\lambda_\tau = \sum_{i=1}^{NL} \lambda_i$ is the expected number of activations per frame in the simulated network. λ_i is the probability of activations per frame.

If the simulated network contains NL links, then $X_{NL} = X_{NL} + X^1_{NL-1}$ is the expected number of activations per minute for NL links. λ_τ is then derived to be

$$\lambda_\tau = X^1_{NL} \cdot \frac{1}{\text{Frame rate}}$$

λ_τ = Total number of activations per frame.

The X/lambda table RXL array produces the message length, with selection dependent upon a random number from 0 to 1. This occurs when the link is activated. The computation is based upon the exponential probability density function for a random variable X having mean value λ :

$$F(X) = \frac{1}{\lambda} e^{-\frac{X}{\lambda}}$$

The random number is generated by a PL1 subroutine RAND (ISEED, RD), where RD is the output random number. ISEED is the input seed, and a new value is returned as the output seed. Initially, ISEED is equal to 1, and this may be reset by the console or file initialization routine in the beginning of this program.

During transmission between the user and the NCS, messages may be lost or garbled. This transmission interference probability is input from data file ID. The PEBF (*,1) array is for transmission from the user to the NCS. PEBF (*,2) is from the NCS to the user.

There are 16 classes of users. Each of these classes has a probability of being one of the eight priorities. These probabilities are read in from input file IA and then accumulated from highest to lowest (8 to 1), with the lowest having a sum of 1.00. When the link is activated, a random number is compared to XPRT (IPG, J), where IPG is the user class or priority group and J equals 8 to 1 for the priorities.

Any initialization printouts are requested by INIT and printed via subroutine IPRNT. The initialization of the simulation ends with the zeroing of the IPBF buffer array.

3.4.4 Subroutine SMA (Simulation Activation)

Subroutine SMA is executed once per frame. The number of activations for each frame is computed and recorded. The links to be activated are then determined.

This program increments the frame counter, calls SNC for messages from the NCS, and determines how many links are to be activated using the URN generator and RS, which is the accumulative Poisson activation probability array.

If the system is saturated, no new activations are initiated. The links that will be activated are selected with the URN generator and XINPT, which is the accumulative activation probability of each link.

The program checks the state of the user link. If it is not in state 0, the program disregards activation procedures and goes to the next link.

A link is activated by setting the user to state 1. The user table ISBF is then established for that link as follows:

ISBF (NPT, 1) = 0 validation count
ISBF (NPT, 2) = INPT (NFLG, 2) MWHO
ISBF (NPT, 3) = Priority using XPRT and INPT (NFLG, 3)
ISBF (NPT, 4) = 1 state
ISBF (NPT, 5) = INPT (NFLG, 4) slots per link
ISBF (NPT, 6) = INPT (NFLG, 5) conference call address
ISBF (NPT, 8) = 0 message count
ISBF (NPT, 10) = INPT (NFLG, 6) mean message length
ITIM (NPT = INPT (NFLG, 6)

The activation statistics are incremented as necessary, and any desired printouts are provided.

3.4.5 Subroutine SMC (Simulation Control Program)

This subroutine is the main program of the simulations. It is entered once per frame and processes all devices, first by messages from the NCS and then by user state.

When entered, the subroutine checks for a system preempt. If there are preempt conditions, the preempt subroutine is called, final messages are printed, and processing is completed.

The program steps through each user via a counter NPT which is from 1 to NUS. Messages from the NCS are in the following categories according to code number:

1 - Assignment

2 - Disconnect

3 - Disconnect - tau delay

4 - ARQ

6 - Request status

10 - Sending message reply to NCS until F = NST

After processing NCS messages, the program proceeds according to the state of the user which is stored in ISBF (NPT, 4).

STATE 1 - Request for channel. The user is newly activated, and a request for a channel is sent to the NCS. The state then becomes 2, and the next NPT is processed.

STATE 2 - Check if message transmitted NST times. If the message transmitted is less than NST times, the message is re-sent via the SRC subroutine. If the message is already sent NST times, the state becomes 3 and the program goes to state 3.

STATE 3 - User waiting for channel assignment. If the assignment message is received from NCS, ITIM (NPT) is calculated as 2 + RTD • FR + 2, and the state of NPT becomes 6.

If no assignment is given, ITIM is incremented until it becomes the maximum waiting time. When the maximum time is reached, the state becomes 4. In both cases control goes to the next NPT.

STATE 4 - Request position in assignment queue. A message is sent to NCS to request the NPT position in the assignment queue. After the message is sent NST times, the state becomes 5. The next NPT is processed.

STATE 5 - Decision after knowing position in assignment queue. First, a message from the NCS with position is searched for. If the request is not given, ITIM is decremented by 1. If ITIM equals 0, a random number is

generated and compared to PR for a decision to return to state 3 or state 0. If the position is received, it is compared to MLQ (maximum queue length). If less than MLQ, NPT goes to state 1. Otherwise, RN is generated and compared with PR as stated above. The next NPT is then processed.

STATE 6 - Handshaking state. The users remain in this state until ITIM equals 0. If ITIM is greater than 0, ITIM is decremented by 1 each frame until it reaches 0.

When ITIM equals 0, the state of the receiver or MWHO is checked for busy or nonbusy status. If MWHO is in state 0 or 13, the handshaking is completed, and the state of NPT is 10 and of MWHO is 9. Message length is computed and inserted in ITIM (NPT).

If MWHO is busy, the waiting queue is checked for openings or for someone waiting with lower priority. The state of NPT is then 7.

If no openings occur in queue, the call is dropped and the state becomes 8.

STATE 7 - Waiting in queue. The state of MWHO is checked. If MWHO is still busy, ITIM is decreased by 1. If ITIM is equal to 0, then the channel is relinquished, MWHO queue is decreased by NPT, and the state of NPT becomes 12.

A deadlock condition occurs when NPT is waiting for MWHO and MWHO is waiting for NPT. When this happens, MWHO relinquishes channel and queue, and becomes state 13.

If MWHO is not busy, the first user in queue is compared to NPT. If NPT is not first, decrease ITIM and go to the next NPT. If NPT is first in MWHO queue, the queue is updated and NPT state becomes 10 and MWHO 9.

STATE 8 - Dropped call. This NPT is added to the statistics and the state becomes 12. The channel is relinquished in the next frame.

STATE 9 - Receiver of call. The NPT queue is checked for a call with higher priority. If there is no higher priority call, the queue goes to the next NPT.

If there is a higher priority call, the present call is canceled to MWHO. If MWHO has someone in queue, MWHO's state is 13; otherwise the MWHO state is 12 and the channel is relinquished. In either case a dropped call is added to the statistics, and the NPT status is 13.

STATE 10 - Check for conference call. If ISBF (NPT, 6) is equal to 0, there is no conference call and the NPT state becomes 11 and control goes to state 11.

If there is a conference call, each participant's state is compared to busy and nonbusy states. States 1, 2, 3, 6, 8, 10, 11, and 12 continue in the present mode.

Each busy participant's present call is compared to the priority of the conference call. If the present call is higher, it is not in conference call.

If conference call is higher in priority, the callers with states 4, 5, 7, 9, 13, and 14 drop their present calls and enter the conference call. Their states become 14. The state of NPT is then 11, and the next NPT is processed.

STATE 11 - User transmitter of call. If there is no one in the queue with a higher priority call, ITIM is decreased by 1 and goes to the next NPT. If ITIM is equal to 0, the call is over and the state becomes 12. Statistics for completed call is updated and NPT continues to state 12.

If there is a higher priority call, this call is dropped and the NPT state becomes 12.

STATE 12 - Relinquishing channel. A message of relinquishment is sent to NCS via subroutine SRC until NST times. If it was a conference call, all participants are released to state 0 or 13, if there is a call in their queue.

STATE 13 - Acknowledge call in queue. This state acknowledges a call in queue. No action is taken by the NPT, but rather by the caller when in state 6 or 7. The program does check to see if there still is a call in the queue. If there is not, the state reverts to 0.

STATE 14 - Conference call third party. This state is for a participant in a conference call who is not one of the two main callers. When all users have been updated for this frame, any output statements requested are printed, and control returns to the main program.

3.4.6 Subroutine STTAT (Statistical Tables)

This subroutine prints the statistical tables accumulated in MSTAT (12, 12, 8). There are 12 tables for 12 slot types and eight priorities. This subroutine is invoked at the end of the run, when there is a preempt, or when periodic statistics are requested.

Average slot occupancy is computed as total slot occupancy times a constant. The constant is calculated at 100/(number of frames times the number of slots per frame). The main loop is for each of the 12 tables. The totals for slots and for priority are computed before the titles are printed.

After the headings and statistics are printed in tabular form, the statistics for the averages are saved. When I equals 8 or 9, the statistics are averaged for "average waiting time for channel" and "average waiting time for handshake." At the end of the subroutine, these statistics are returned to the values they had before entering the subroutine. This includes the statistics for "average slot occupancy."

The 12 categories are (1) activations, (2) completed calls, (3) bumped calls, (4) bumped requests from NCS, (5) channel assignments, (6) completed handshakes, (7) ARQ, (8) average waiting time for channel, (9) average waiting time for handshake, (10) average slot occupancy, (11) maximum wait/ no assignment, and (12) system violations.

3.4.7 Subroutine SRC (NPP, MMX, I4) (Message to NCS)

This subroutine assembles the message from the user to the NCS and sends the message ST times. The interface buffer IPBF is filled using either the user array ISBF or IOBF.

For each user that transmits a message, this routine is called once per frame, but the message is written into all the slots allotted for that user per frame. Once a message is initiated, it is re-sent NST times.

3.4.8 Subroutine SVC (Simulation Validity Control)

This subroutine is used to validate entries in IPBF in the random access mode only.

At the end of SMC, SVC is called to check for all overwrites and then to validate message transmission of those without overwrites.

All messages have I equal to 0 when entering this routine. I = IPBF (LSLOT, 7) = 0.

First the subroutine checks for H = IPBF (LSLOT, 6) which is the overwrite condition. If H is greater than 0, there are overwrites and the message is not valid.

If a valid message has previously been sent, another is not sent again. A random number is generated and compared with the PEBF array for transmission errors.

If the message is valid, the validity count in inserted in IPBF (LSLOT, 7).

3.4.9 Subroutine SNC

This subroutine converts messages from the NCS to the user into a numbered code. The program searches the IPBF array for validated messages once per frame. A message is valid if IPBF (user, 7) $\geq$ NSS and IPBF (user, 5) = NST.

The messages are inserted into ISBF (user, 9) with the following codes:

0 - None

1 - Assignment

2 - Disconnect no delay

3 - Disconnect tau delay

4 - ARQ

6 - Request status

-XX - Position in assignment queue

After decoding the messages from the NCS, the program initializes the IPBF array.

3.4.10 Subroutine QUEUE (MT, MF)

This subroutine drops a user from the waiting queue of another user. It is called from various places in the SMC subroutine. Whenever a call is dropped (due to time delay or a higher priority call), QUEUE is called.

The entering parameters are MT, which is the user whose queue is being reduced, and MF, which is the user who is dropping the call.

The program searches for MF in ITBF (MT, I, 1) and then moves all users beyond that point forward one place in the queue.

3.4.11 Subroutine IPRNT

This subroutine prints the initialization variables and tables as produced in subroutine SIMIN. If INIT is equal to 0, this subroutine is not entered.

The output is structured into four tables: the TDMA simulator input, the linkage table, the cummulative Poisson activation probabilities, and the X/lambda table.

3.4.12 Subroutine PREEMP

This subroutine is called when there is a system preempt. It prints out a message that a preempt condition exists at frame NFR.

The subroutine then calls STTAT to print the statistics of the run to that point. When control returns, PREEMP stops the run. The run length in seconds before a preempt is an input value to SIMIN.

3.4.13 Subroutine RAND (I, R)

This subroutine provides the uniform random number by calling the system uniform random number generator program random-$uniform (i, r). The system subroutine is in PL1, so RAND is needed to interface the FORTRAN subroutines with the PL1 system subroutine.

R is the random number generated, and I is the input seed to the subroutine when called and the output for the next seed when returned. R is a floating-point variable from 0 to 1.

4. SAMPLE RUN

The following printout contains a run of 1,000 seconds or 1,000 frames at the frame rate of one frame per second. There are 30 users and 60 links in the network. The links require four or eight slots for a channel. The priorities are a mixture of all eight possible priorities. Two types of console commands are inserted. On frame 50, a status request from user 17 is included. In addition, the run stops every 100 frames and prompts the operator as follows:

- Inquire about a user (by user ID)
- Inquire about a channel (by channel ID)
- Enter a console command.

Every 100 frames, as many of these console requests may be inserted as desired.

Pages 82 - 86 are printouts of tables which were previously stored and are used in this run. Information from these tables must be included during the initialization phase of the simulation.

"ISBF" on page 82 is a linking slot assignment file. "PEBF" on page 83 specifies the error rate from user to NCS and from NCS to user. "DIDC" on pages 84 - 85 specifies the mean message length, activation rate, priority, burst rate and data rate of each link. File DINIT on page 86 specifies a large number of run parameters as indicated on pages 87 and 88.

```
io_call attach file01 vfile_ didcy.12848
r 951 0.503 4.866 97

io_call attach file02 vfile_ dpebf
r 951 0.118 0.416 27

o#io_call attach file03 vfile_ isbf.40
r 951 0.123 0.730 35

io_call attach file10 vfile_ dinit2.exp
r 952 0.112 0.826 36



print isbf.40

        isbf.40     07/20/77  0957.2 edt Wed
```

3401	3501	3601	3701	3801	3901	4001	4101
4201	4301	4401	4501	4601	4701	4801	4901
5001	5101	5201	5301	5401	5501	5601	5701
5801	5901	6001	6101	6201	6301	6401	6501
6601	6701	6801	6901	7001	7101	7201	7301
7401	7501	7601	7701	7801	7901	8001	8101

```
r 957 0.656 0.566 23
```

```
print dprebf

                    dprebf     07/20/77  0957.9 edt Wed

 .02 .02
 .30 .30
 .05 .05
 .10 .10
 .25 .02
 .02 .25
 .10 .10
 .15 .15
 .05 .05
 .02 .30
 .15 .10
 .10 .15
 .05 .05
 .25 .02
 .05 .05
 .02 .05
 .05 .02
 .03 .03
 .27 .23
 .11 .12
 .01 .05
 .03 .10
 .15 .14
 .22 .16
 .17 .18
 .01 .05
 .02 .06
 .02 .03
 .10 .02
 .05 .05

r 957 0.405 0.830 39
```

```
print didcy.12848
                     didcy.12848          07/20/77  1000.9 edt Wed

        1      2      120.     .030    1   614400.   614400. 19200.    1 0
        1     10      120.     .030    2   614400.   614400. 19200.    2 0
        1     13      120.     .030    3   614400.   614400. 19200.    1 0
        1     14      120.     .030    4   614400.   614400. 19200.    1 0
        1     15      120.     .030    5   614400.   614400. 19200.    1 0
        2      3      120.     .030    6   614400.   614400. 19200.    2 0
        2      4      120.     .030    7   614400.   614400. 19200.    2 0
        2      6      120.     .030   16   614400.   614400. 19200.    1 0
        2      8      120.     .030    8   614400.   614400. 19200.    2 0
        3      1      120.     .030    8   614400.   614400.   19200.  1 0
        3      7      120.     .030    8   614400.   614400. 19200.    2 2
       10     12
        3     12      120.     .030   15   614400.   614400. 19200.    1 3
        1      4        8
        4      5      120.     .030   10   614400.   614400. 19200.    2 0
        4      9      120.     .030    9   614400.   614400. 19200.    1 3
        1     11       15
        4     13      120.     .030    5   614400.   614400. 19200.    2 0
        5     10      120.     .030   14   614400.   614400. 19200.    1 0
        5     11      120.     .030   13   614400.   614400. 19200.    2 0
        5     14      120.     .030   12   614400.   614400. 19200.    2 0
        6      1      120.     .030   15   614400.   614400. 19200.    2 2
        3      4
        6      3      120.     .030    6   614400.   614400. 19200.    2 0
        6     15      120.     .030   11   614400.   614400. 19200.    2 0
        7      2      120.     .030    2   614400.   614400. 19200.    2 3
        9     11       15
        7      4      120.     .030   10   614400.   614400. 19200.    1 3
        1     11       12
        8      5      120.     .030   11   614400.   614400. 19200.    2 0
        8     10      120.     .030    1   614400.   614400. 19200.    2 0
        8     13      120.     .030    5   614400.   614400. 19200.    1 0
        9      7      120.     .030    3   614400.   614400. 19200.    2 1
        5
       10      9      120.     .030    3   614400.   614400. 19200.    2 0
       11      2      120.     .030   10   614400.   614400. 19200.    1 3
        7      8        9
       11      5      120.     .030   15   614400.   614400. 19200.    1 0
       11     14      120.     .030    9   614400.   614400. 19200.    1 3
        1      3        5
       12     11      120.     .030   16   614400.   614400. 19200.    2 0
       12     14      120.     .030    3   614400.   614400. 19200.    1 0
       13      2      120.     .030    7   614400.   614400. 19200.    1 0
       13      6      120.     .030    5   614400.   614400. 19200.    1 0
       14      3      120.     .030    4   614400.   614400. 19200.    2 2
        6      9
       14      7      120.     .030    2   614400.   614400. 19200.    1 3
        1      5        8
```

15	1	120.	.030	7	614400.	614400.	19200.	1	0
15	4	120.	.030	1	614400.	614400.	19200.	1	1
8									
15	10	120.	.030	4	614400.	614400.	19200.	1	0
16	10	120.	.030	8	614400.00	614400.00	19200.00	1	0
16	18	120.	.030	8	614400.00	614400.00	19200.00	1	0
17	1	120.	.030	8	614400.00	614400.00	19200.00	1	0
18	5	120.	.030	8	614400.00	614400.00	19200.00	1	0
18	15	120.	.030	8	614400.00	614400.00	19200.00	1	0
19	4	120.	.030	8	614400.00	614400.00	19200.00	1	0
20	18	120.	.030	8	614400.00	614400.00	19200.00	1	0
20	25	120.	.030	8	614400.00	614400.00	19200.00	1	0
21	30	120.	.030	8	614400.00	614400.00	19200.00	1	0
22	1	120.	.030	8	614400.00	614400.00	19200.00	1	0
23	27	120.	.030	8	614400.00	614400.00	19200.00	2	0
23	30	120.	.030	8	614400.00	614400.00	19200.00	2	0
24	17	120.	.030	8	614400.00	614400.00	19200.00	2	0
25	19	120.	.030	8	614400.00	614400.00	19200.00	2	0
26	2	120.	.030	8	614400.00	614400.00	19200.00	2	0
27	12	120.	.030	8	614400.00	614400.00	19200.00	2	0
28	16	120.	.030	8	614400.00	614400.00	19200.00	2	0
29	15	120.	.030	8	614400.00	614400.00	19200.00	2	0
30	10	120.	.030	8	614400.00	614400.00	19200.00	2	0
30	20	120.	.030	8	614400.00	614400.00	19200.00	2	0

r 1001 1.524 0.520 45

```
print dinit2.exp

                  dinit2.exp          07/07/77  1607.1 edt Thu


   2
  30
  60
   3    5
1.
 128
2.
   2
   6
1200.
1000.
 .25
  25
   1
   2
  25
  1.
 .75
   1
   0
  -1
5.
.5
 2.
  32
  96
   7
 10.
  .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1
  .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1
  .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1
  .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1
  .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1
  .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1    .1
  .2    .2    .2    .2    .2    .2    .2    .2    .2    .2    .2    .2    .2    .2    .2    .2
  .2    .2    .2    .2    .2    .2    .2    .2    .2    .2    .2    .2    .2    .2    .2    .2
   1
100.
   0
   0.
   1
   1
  50
   4
   5
   8
  17


r 1607 0.544 0.078 10 level 2, 12
```

```
main
 input ia,ib,ic,id,ie,ih
 10  6  1  2  3  6
 mode of linkage access ; assignment=1, polling=2, random=3.
 number of users in network.
 number of links in network.
 strategy:nns out of nst transmissions.
 nss and nst=
 frame rate
 slots/frame
 disconnect delay in seconds
 disconnect strategy:1=RFR,2=forced
 queue length/device (max of 6)
 system preempt (seconds)
 system run length in seconds
 decision threshold for channel relinquishment
 maximum delay to wait for channel
 maximum queue length before giving up assignment wait
 maximum transmission delay
 maximum delay in destination queue
 probability of giving up after max wait
 factor for saturated system
 seed for urn generator
 frame-to-frame printouts? (1=y,0=n)
 initialization printouts? (1=y,0=n)
 waiting time factor for position in queue
 factor to shorten call before disconnect
 round trip delay
 no. of ncs linking slots
 number of active linking slots
 number of different channels
 max delay user to wait for assignment in NCS queue
 xprt table 8 rows with 16 probabilities
 input each priority on separate line for 8 lines of input
```

```
                    tdma simulator
                    **************

mode of linkage access                                  2
no. of users                                           30
no. of links                                           60
strategy:      3 out of     5 transmissions must be valid.
frame rate                                             1.000
slots/frame                                          128
disconnect delay (sec)                                 2.000
queue length/device                                     6
disconnect by (1=rfr or 2=forced)                      2
system preempt                                      1200.000
system run length                                   1000.000
decision threshold for channel relinquishment          0.250
user delays: max for assignment                        25
             max in queue                              25
             probability of relinguishing              1.000
             max transmission delay                     2
             max delay in ncs queue                    10
factor for saturated system                            0.750
waiting factor for queue                               5.000
factor to shorten call for disconnect                  0.500
round trip delay                                        1
no. of linking slots for ncs                           32
no. of linking slots active                            96
number of different channels                            7
max queue position to wait for channel                  1


 rlams=    0.180000e+01          rlamt=    0.180000e+01
```

AD-A047 615 COMPUTER SCIENCES CORP FALLS CHURCH VA F/G 17/2.1
TDMA SATCOM SYSTEM SIMULATION.(U)
NOV 77 G FRENKEL, P KELL F30602-76-C-0247
UNCLASSIFIED RADC-TR-77-354 NL

2 OF 2
AD
A047615

END
DATE
FILMED
1-78
DDC

```
accum.poisson act. prob:         0        0.165299e+00
accum.poisson act. prob:         1        0.462837e+00
accum.poisson act. prob:         2        0.730621e+00
accum.poisson act. prob:         3        0.891292e+00
accum.poisson act. prob:         4        0.963593e+00
accum.poisson act. prob:         5        0.989622e+00
accum.poisson act. prob:         6        0.997431e+00
accum.poisson act. prob:         7        0.999438e+00
accum.poisson act. prob:         8        0.999890e+00
accum.poisson act. prob:         9        0.999981e+00
accum.poisson act. prob:        10        0.999997e+00
accum.poisson act. prob:        11        0.100000e+01
accum.poisson act. prob:        12        0.100000e+01
accum.poisson act. prob:        13        0.100000e+01
accum.poisson act. prob:        14        0.100000e+01


x/lamda=          0.100
x/lamda=          0.105
x/lamda=          0.223
x/lamda=          0.357
x/lamda=          0.511
x/lamda=          0.693
x/lamda=          0.916
x/lamda=          1.204
x/lamda=          1.609
x/lamda=          2.303
x/lamda=          3.000


end of simin
```

Summary of NCS initialization data :

Simulation control is entered every 100.000 seconds
which corresponds to 100 frames

There are no permanently assigned linking channels

There are no permanently assigned, non-linking channels

User status is not requested periodically

There are 1 console commands as follows :

Command	User	Priority	Type	Execution frame
STA	17	8	5	50

```
frame    0



NCS processing is completed for frame     100



Do you wish to inquire about a user ?  (1=y,0=n)
   1

User's ID :
  27

A channel is assigned

Channel assignment number =    24
Channel type =  5

Priority = 8



Do you wish to inquire about a user ?  (1=y,0=n)
   1

User's ID :
   6

A channel is assigned

Channel assignment number =     4
Channel type =  5

Priority = 8
```

```
Do you wish to inquire about a user ?  (1=y,0=n)
   1

User's ID :
   9

A channel is assigned

Channel assignment number =    1
Channel type =  5

Priority = 6



Do you wish to inquire about a user ?  (1=y,0=n)
   1

User's ID :
   7

No channel is assigned to this user

No request is waiting to be processed



Do you wish to inquire about a user ?  (1=y,0=n)
   1

User's ID :
   1

A channel is assigned

Channel assignment number =    8
Channel type =  5

Priority = 7
```

```
Do you wish to inquire about a user ?  (1=y,0=n)
   1

User's ID :
  30

A channel is assigned

Channel assignment number =    18
Channel type =  5

Priority = 5



Do you wish to inquire about a user ?  (1=y,0=n)
   1

User's ID :
   2

A channel is assigned

Channel assignment number =     9
Channel type =  4

Priority = 4



Do you wish to inquire about a user ?  (1=y,0=n)
   0



Do you wish to inquire about a particular channel ?  (1=y, 0=n)
   1

Channel assignment number :
  18

Channel type :
   5

This channel is occupied by user   30
```

```
Do you wish to inquire about a particular channel ?  (1=y, 0=n)
   1

Channel assignment number :
  27

Channel type :
   4

This channel is vacant




Do you wish to inquire about a particular channel ?  (1=y, 0=n)
   0



Do you wish to enter a console command ?  (1=y,0=n)
   1

(i5 format)
------------
User's id
   7

enter 1=RFA; 2=RFR; 3=RFP; 4= user status information
   1

Channel type
   4

Priority
   8



Do you wish to enter a console command ?  (1=y,0=n)
   0
frame  100




NCS processing is completed for frame    200
```

```
Do you wish to inquire about a user ?  (1=y,0=n)
   1

User's ID :
   7

A channel is assigned

Channel assignment number =    12
Channel type =  5

Priority = 5



Do you wish to inquire about a user ?  (1=y,0=n)
   1

User's ID :
   7

A channel is assigned

Channel assignment number =    12
Channel type =  5

Priority = 5



Do you wish to inquire about a user ?  (1=y,0=n)
   0



Do you wish to inquire about a particular channel ?  (1=y, 0=n)
   1

Channel assignment number :
  12

Channel type :
   5

This channel is occupied by user    7
```

```
Do you wish to inquire about a particular channel ?  (1=y, 0=n)
   0


Do you wish to enter a console command ?  (1=y,0=n)
   0
frame  200



NCS processing is completed for frame     300



Do you wish to inquire about a user ?  (1=y,0=n)
   0


Do you wish to inquire about a particular channel ?  (1=y, 0=n)
   0


Do you wish to enter a console command ?  (1=y,0=n)
   0
frame  300



NCS processing is completed for frame     400
```

```
Do you wish to inquire about a user ?  (1=y,0=n)
    0



Do you wish to inquire about a particular channel ?  (1=y, 0=n)
    0



Do you wish to enter a console command ?  (1=y,0=n)
    0
frame  400



NCS processing is completed for frame    500



Do you wish to inquire about a user ?  (1=y,0=n)
    0



Do you wish to inquire about a particular channel ?  (1=y, 0=n)
    0



Do you wish to enter a console command ?  (1=y,0=n)
    0
frame  500
```

```
NCS processing is completed for frame    600



Do you wish to inquire about a user ?  (1=y,0=n)
    0


Do you wish to inquire about a particular channel ?  (1=y, 0=n)
    0


Do you wish to enter a console command ?  (1=y,0=n)
    0
 frame  600



NCS processing is completed for frame    700



Do you wish to inquire about a user ?  (1=y,0=n)
    0


Do you wish to inquire about a particular channel ?  (1=y, 0=n)
    0
```

```
Do you wish to enter a console command ?  (1=y,0=n)
   0
frame  700


NCS processing is completed for frame    800


Do you wish to inquire about a user ?  (1=y,0=n)
   0


Do you wish to inquire about a particular channel ?  (1=y, 0=n)
   0


Do you wish to enter a console command ?  (1=y,0=n)
   0
frame  800


NCS processing is completed for frame    900
```

```
Do you wish to inquire about a user ?  (1=y,0=n)
    0

Do you wish to inquire about a particular channel ?  (1=y, 0=n)
    0

Do you wish to enter a console command ?  (1=y,0=n)
    0
frame  900
```

STATISTICS

Elapsed Time 1000 Frames 1000.00 Seconds

Activations

SLOT SIZE (2**n)

PRIORITY		0	1	2	3	4	5	6	7	8	9	10	11	T
P	1	0	0	12	12	0	0	0	0	0	0	0	0	24
R	2	0	0	16	10	0	0	0	0	0	0	0	0	26
I	3	0	0	12	15	0	0	0	0	0	0	0	0	27
O	4	0	0	8	15	0	0	0	0	0	0	0	0	23
R	5	0	0	14	8	0	0	0	0	0	0	0	0	22
I	6	0	0	21	15	0	0	0	0	0	0	0	0	36
T	7	0	0	34	29	0	0	0	0	0	0	0	0	63
Y	8	0	0	17	23	0	0	0	0	0	0	0	0	40
	T	0	0	134	127	0	0	0	0	0	0	0	0	261

Completed Calls

SLOT SIZE (2**n)

PRIORITY		0	1	2	3	4	5	6	7	8	9	10	11	T
P	1	0	0	1	0	0	0	0	0	0	0	0	0	1
R	2	0	0	4	0	0	0	0	0	0	0	0	0	4
I	3	0	0	4	2	0	0	0	0	0	0	0	0	6
O	4	0	0	0	1	0	0	0	0	0	0	0	0	1
R	5	0	0	3	0	0	0	0	0	0	0	0	0	3
I	6	0	0	4	2	0	0	0	0	0	0	0	0	6
T	7	0	0	11	7	0	0	0	0	0	0	0	0	18
Y	8	0	0	10	11	0	0	0	0	0	0	0	0	21
	T	0	0	37	23	0	0	0	0	0	0	0	0	60

Bumped Calls

SLOT SIZE (2**n)

PRIORITY		0	1	2	3	4	5	6	7	8	9	10	11	T
P	1	0	0	0	1	0	0	0	0	0	0	0	0	1
R	2	0	0	3	1	0	0	0	0	0	0	0	0	4
I	3	0	0	1	0	0	0	0	0	0	0	0	1	2
O	4	0	0	2	1	0	0	0	0	0	0	0	0	3
R	5	0	0	3	1	0	0	0	0	0	0	0	3	7
I	6	0	0	2	2	0	0	0	0	0	0	0	2	6
T	7	0	0	3	7	0	0	0	0	0	0	0	0	10
Y	8	0	0	1	2	0	0	0	0	0	0	0	0	3
	T	0	0	15	15	0	0	0	0	0	0	0	6	36

Bump Requests from NCS

SLOT SIZE (2**n)

		0	1	2	3	4	5	6	7	8	9	10	11	T
P	1	0	0	0	3	0	0	0	0	0	0	0	0	3
R	2	0	0	0	3	0	0	0	0	0	0	0	0	3
I	3	0	0	0	4	0	0	0	0	0	0	0	0	4
O	4	0	0	0	7	0	0	0	0	0	0	0	0	7
R	5	0	0	0	2	0	0	0	0	0	0	0	0	2
I	6	0	0	0	3	0	0	0	0	0	0	0	0	3
T	7	0	0	0	5	0	0	0	0	0	0	0	0	5
Y	8	0	0	0	0	0	0	0	0	0	0	0	0	0
	T	0	0	0	27	0	0	0	0	0	0	0	0	27

Channel Assignments

SLOT SIZE (2**n)

		0	1	2	3	4	5	6	7	8	9	10	11	T
P	1	0	0	10	5	0	0	0	0	0	0	0	0	15
R	2	0	0	12	6	0	0	0	0	0	0	0	0	18
I	3	0	0	10	9	0	0	0	0	0	0	0	0	19
O	4	0	0	4	11	0	0	0	0	0	0	0	0	15
R	5	0	0	11	5	0	0	0	0	0	0	0	0	16
I	6	0	0	18	7	0	0	0	0	0	0	0	0	25
T	7	0	0	30	20	0	0	0	0	0	0	0	0	50
Y	8	0	0	14	18	0	0	0	0	0	0	0	0	32
	T	0	0	109	81	0	0	0	0	0	0	0	0	190

Completed Handshakes

SLOT SIZE (2**n)

PRIORITY	0	1	2	3	4	5	6	7	8	9	10	11	T
1	0	0	2	1	0	0	0	0	0	0	0	0	3
2	0	0	6	0	0	0	0	0	0	0	0	0	6
3	0	0	5	4	0	0	0	0	0	0	0	0	9
4	0	0	1	3	0	0	0	0	0	0	0	0	4
5	0	0	3	2	0	0	0	0	0	0	0	0	5
6	0	0	10	5	0	0	0	0	0	0	0	0	15
7	0	0	17	15	0	0	0	0	0	0	0	0	32
8	0	0	11	13	0	0	0	0	0	0	0	0	24
T	0	0	55	43	0	0	0	0	0	0	0	0	98

ARQ

SLOT SIZE (2**n)

PRIORITY	0	1	2	3	4	5	6	7	8	9	10	11	T
1	0	0	0	1	0	0	0	0	0	0	0	0	1
2	0	0	0	1	0	0	0	0	0	0	0	0	1
3	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0	0	0	0	1
5	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	1	1	0	0	0	0	0	0	0	0	2
7	0	0	1	0	0	0	0	0	0	0	0	0	1
8	0	0	0	0	0	0	0	0	0	0	0	0	0
T	0	0	2	4	0	0	0	0	0	0	0	0	6

Average Waiting Time for Channel (seconds)

SLOT SIZE (2**n)

PRIORITY		0	1	2	3	4	5	6	7	8	9	10	11	T
P	1	0.000	0.000	1.400	2.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.600
R	2	0.000	0.000	1.417	3.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.944
I	3	0.000	0.000	1.000	4.222	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	2.526
O	4	0.000	0.000	1.000	4.636	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	3.667
R	5	0.000	0.000	1.091	3.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.688
I	6	0.000	0.000	1.500	4.143	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	2.240
T	7	0.000	0.000	1.500	6.650	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	3.560
Y	8	0.000	0.000	1.429	4.111	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	2.938
	T	0.000	0.000	1.367	4.543	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	2.721

Average Waiting Time for Handshake (seconds)

SLOT SIZE (2**n)

PRIORITY		0	1	2	3	4	5	6	7	8	9	10	11	T
P	1	0.000	0.000	23.500	25.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	24.000
R	2	0.000	0.000	10.667	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	10.667
I	3	0.000	0.000	10.800	16.500	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	13.333
O	4	0.000	0.000	6.000	17.333	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	14.500
R	5	0.000	0.000	14.000	22.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	17.200
I	6	0.000	0.000	9.400	25.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	14.600
T	7	0.000	0.000	16.941	16.267	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	16.625
Y	8	0.000	0.000	19.727	17.077	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	18.292
	T	0.000	0.000	14.764	18.093	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	16.224

Average Slot Occupancy (Percent)
(Results do not include permanently assigned slots)

SLOT SIZE (2**n)

PRIORITY		0	1	2	3	4	5	6	7	8	9	10	11	T
P	1	0.000	0.000	2.069	4.531	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	6.600
R	2	0.000	0.000	4.459	0.912	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	5.372
I	3	0.000	0.000	1.678	3.737	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	5.416
O	4	0.000	0.000	2.091	2.562	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	4.653
R	5	0.000	0.000	3.053	1.006	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	4.059
I	6	0.000	0.000	3.450	3.381	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	6.831
T	7	0.000	0.000	9.006	11.325	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	20.331
Y	8	0.000	0.000	4.566	10.906	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	15.472
	T	0.000	0.000	30.372	38.362	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	68.734

Maximum Wait for Channel/No Assignment

SLOT SIZE (2**n)

PRIORITY		0	1	2	3	4	5	6	7	8	9	10	11	T
P	1	0	0	2	5	0	0	0	0	0	0	0	0	7
R	2	0	0	3	4	0	0	0	0	0	0	0	0	7
I	3	0	0	1	5	0	0	0	0	0	0	0	0	6
O	4	0	0	4	4	0	0	0	0	0	0	0	0	8
R	5	0	0	3	3	0	0	0	0	0	0	0	0	6
I	6	0	0	3	7	0	0	0	0	0	0	0	0	10
T	7	0	0	4	9	0	0	0	0	0	0	0	0	13
Y	8	0	0	3	5	0	0	0	0	0	0	0	0	8
	T	0	0	23	42	0	0	0	0	0	0	0	0	65

System Violations (ARQ on RFR'S)

SLOT SIZE (2**n)

PRIORITY	0	1	2	3	4	5	6	7	8	9	10	11	T
1	0	0	2	0	0	0	0	0	0	0	0	0	2
2	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0	0	0	1
4	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	3	0	0	0	0	0	0	0	0	0	3
8	0	0	1	0	0	0	0	0	0	0	0	0	1
T	0	0	6	1	0	0	0	0	0	0	0	0	7

APPENDIX A - AN EFFICIENT METHOD OF SIMULATION FOR TIME-SHARING SYSTEMS

An Efficient Method of Simulation for Time-Sharing Systems

GABRIEL FRENKEL, MEMBER, IEEE

***Abstract*—Simulation of requests for service from a large number of low duty cycle independent users is apt to cause a severe computational load. A method is presented for the software implementation of an efficient simulator. It can be applied to the investigation of many time-sharing systems, such as satellite demand assignment (DA), computer time sharing, and traffic routing. More generally, the method generates independent multivariate Poisson distributions when the number of variables is very large and the value of the parameter in each distribution is very small.**

Let there be a satellite system which serves as a means of communications between a large number of users. The traffic on the various user-to-user links has low duty cycle and is characterized for each link by the call activation rate, data rate, and mean call duration. The calls on the link are independently distributed and their occurrences obey the Poisson distribution. The Poisson parameters are widely different for the various links. If all the links were simultaneously active, the demand would outstrip by far the throughput capabilities of the satellite system, hence demand assignment (DA) is used for channel allocation. All the users have permanently allocated signaling channels of extremely low data rate. Whenever a user wishes to communicate with another it asks a network control terminal (NCT) to assign satellite resources to this end. This resource could be fractional satellite power or time slot, depending on the method of multiple access, and is not addressed here. If available, the NCT assigns such a resource commensurate with the characteristics, data rate, and terminal sizes of the particular link to be established. This is accomplished through a fairly complex allocation program at the NCT, which must keep track of channels, active users, priorities, and other system status parameters.

The most expedient means of evaluating the performance of such a system is through simulation. Fig. 1 shows the major elements of the software. The traffic simulator contains the list of all users, their connectivities (links), and the aforementioned link parameters. During each simulation interval T, inactive links are activated probabilistically according to the activation rate, and the identities of activated links are sent to the active link simulator. Here various scenarios are enacted and demands for service or relinquishment of previously allocated facilities are sent to the system simulator, whence in turn assignments or requests for relinquishments are returned. Upon cessation of activity on a link, a deactivation signal (relinquishment) is sent to the traffic simulator from the active link simulator.

Paper approved by the Editor for Communication Systems Disciplines of the IEEE Communications Society for publication without oral presentation. Manuscript received April 14, 1976; revised June 3, 1976. This work was supported in part by the Rome Air Development Center, Griffiss Air Force Base, under Contract F 30602-74-C-0185.

The author is with the Computer Sciences Corporation, Falls Church, VA.

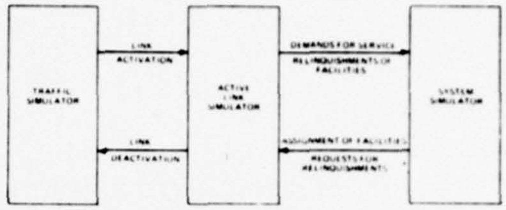


Fig. 1. Simulation of DA.

The configuration shown in Fig. 1 is valid for a much larger class of problems than the specific DA problem presented here as an illustration. Other examples are computer time sharing and traffic routing. In all these, the difficulty is in simulating the activation in the first block of Fig. 1. Let there be n links in the system, with the number of calls activated during the simulation interval T having independent Poisson distributions with parameters $\lambda_1 \cdots \lambda_n$. In practice, T is taken sufficiently small so that an inactive link is activated only once during the simulation interval T with probability λ_i. Let there be associated with each link the index A_j, $j = 1, 2, \cdots, n$, where $A_j = 1$ if the link is active and zero otherwise. The function of the traffic simulator is to change, during each simulator interval, the value of A_j, if $A_j = 0$, from 0 to 1 with probability λ_j. At the beginning of each interval some of the previously active links are deactivated by the active link simulator, i.e., A_j is reset from 1 to 0. The procedure is to be repeated for each subsequent time interval T. The number and identity of active links changes during each interval according to the activations and deactivations caused by the traffic simulator and active link simulator, respectively.

The obvious simulation procedure for the traffic simulator is to generate for the d inactive links, d random variables $x_1, x_2, \cdots, x_d$, linearly and independently distributed from 0 to 1. If $x_i < \lambda_i$, then $A_i = 1$, otherwise $A_i = 0$. Since the number of links to be considered could be very large, this approach would necessitate the generation of a few thousand random variables. A typical example follows.

- number of links = 4000;
- computer cycle time = 1 μs;
- number of computer cycles/uniform random number generation = 50;
- number of computer cycles for a comparison = 8;
- computer time/simulation cycle = $4000 \times 10^{-6} \times (58)$ s = 0.232 s.

The foregoing highlights the problems encountered when simulating a large number of independent, low duty cycle users of widely varying characteristics, sharing a facility. The following procedure will achieve the desired simulation objectives with considerably fewer calculations.

Step 1: Compute the series,

$$t_k = t_{k-1} + \lambda_k, \quad k = 1, 2, \cdots, n; \; t_0 = 0.$$

Step 2: Compute the series $p_0, p_1, \cdots$ where

$$p_m = e^{-\lambda}\frac{\lambda^m}{m!}, \qquad \lambda = t_n = \sum_{j=1}^{n} \lambda_j, \qquad m = 0, 1, 2, \cdots$$

Here p_m is the conditional probability that m links will be activated during the interval, given that all links are inactive at the beginning of the interval.

Step 3: Compute the conditional cumulative distribution

$$s_k = \sum_{i=0}^{k} p_i, \qquad k = 0, 1, \cdots, n$$

Step 4: Generate the random number r_0 uniformly distributed between 0 and 1. Find s_k such that

$$s_{k-1} \leqslant r_0 < s_k$$

k would be the number of new activations during the interval T if all the links were inactive at the beginning of the interval.

Step 5: Generate a random number r uniformly distributed between 0 and 1. Find j such that

$$t_{j-1} < r\lambda \leqslant t_j$$

Step 6: Repeat procedure in Step 5, k times, thus generating the random variables $y_1, y_2, \cdots, y_n$, where

$$y_1 + y_2 + \cdots + y_n = k$$

and y_j is the number of times j was obtained after performing Step 5 k times.

Step 7: Set $A_j = 1$ if $y_j \neq 0$ and A_j was equal to zero previously.

Step 8: Change some of the values of A_j from 1 to 0 according to the deactivation input from the active link simulator.

Step 9: Repeat Steps 4 through 7 for the next interval T.

The number of random numbers generated during the simulation interval T equals $k + 1$, which in most practical cases represents a reasonably small computational load.

It can be readily demonstrated that the procedure satisfies the simulation objectives. The joint conditional distribution of the random variables $y_1, y_2, \cdots, y_n$ given k is multinomial

$$p(y_1, y_2, \cdots, y_n/k) = \frac{k!}{y_1! y_2! \cdots y_n!}\left(\frac{\lambda_1}{\lambda}\right)^{y_1} \cdot \left(\frac{\lambda_2}{\lambda}\right)^{y_2} \cdot \cdots \cdot \left(\frac{\lambda_n}{\lambda}\right)^{y_n}$$

where the sample space includes all points for which

$$\sum_{i=1}^{n} y_i = k$$

The joint distribution of $y_1, y_2, \cdots, y_n$ is given by

$$p(y_1, y_2, \cdots, y_n) = p(y_1, y_2, \cdots, y_n/k)p_k = \prod_{i=1}^{n}\left(\frac{\lambda_i}{\lambda}\right)^{y_i}\frac{e^{-\lambda}}{y_i!}\lambda^k = \prod_{i=1}^{n}\frac{\lambda_i^{y_i}}{y_i!}e^{-\lambda_i}$$

The n random variables are independent and have the Poisson distribution with parameters λ_i, $i = 1, 2, \cdots, n$.

A numerical example follows (it will be assumed that the computer has the cycle time, and computer cycles for random number generation and comparisons as in the previous example):

number of links = 4000;
$\lambda = 1$ = mean value of k;
number of cycles:

- Step 1 = 0; Step 2 = 0; Step 3 = 0; — The first three steps are part of the input table; the values are computed prior to the simulation runs.
- Step 4: One random number generation and one comparison = 58 cycles;
- Step 5 and 6: On the average, one random number is generated and a table search is performed. Assuming 20 cycles for each step in the table search: number of cycles = 50 + 20 $\log_2 n$ = 290 cycles;
- Steps 7 and 8: 20 cycles.

Computer time/simulation cycle = (58 + 290 + 20) × 10^{-6} s = 368 μs. This is considerably smaller than the 0.232 s obtained previously.

From the foregoing, it may be concluded that the method presented here could reduce the computer CPU time by orders of magnitude, compared to more conventional means. The actual savings will depend on the computer parameters and the parameters of the traffic being simulated.

REFERENCES

1. TDMA SATCOM System Simulation Software Documentation, Prepared by Computer Sciences Corporation, May 1977.

2. TDMA Network Control Study, Prepared by Computer Sciences Corporation, August 1973, RADC-TR-73-270. (AD767218)

3. Statement of Work for TDMA SATCOM System Simulation, Prepared by Rome Air Development Center, October 1975, PRNDC-6-2410.

MISSION
of
Rome Air Development Center

RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C^3) activities, and in the C^3 areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.

AMERICAN REVOLUTION BICENTENNIAL 1776-1976